

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 12/14/89
Original Closeout Started *****

Project No. E-21-670_____ Center No. R6071-0A0_____

Project Director CLEMENTS M A_____ School/Lab EE_____

Sponsor US DEPT OF DEFENSE/MARYLAND PROCUREMENT OFF_____

Contract/Grant No. MDA904-85-K-0005_____ Contract Entity GTRC

Prime Contract No. _____

Title RESEARCH IN DIGITAL SIGNAL PROCESSING_____

Effective Completion Date 890331 (Performance) 890430 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	_____
Final Report of Inventions and/or Subcontracts	Y	_____
Government Property Inventory & Related Certificate	Y	_____
Classified Material Certificate	N	_____
Release and Assignment	Y	_____
Other _____	N	_____

Subproject Under Main Project No. A-3215_____

Continues Project No. _____

Distribution Required:

Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Management	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
OCA/CSD	N
Other _____	N
_____	N

NOTE: Final Questionnaire sent to PDPI.

PROJECT ADMINISTRATION DATA SHEET☒ ORIGINAL ☐ REVISION NO. _____Project No. E-21-670 (R-6071-0A0) GTRC/~~AKK~~ DATE 12/ 12/ 85Project Director: M. Clements MSL School MS EESponsor: Maryland Procurement OfficeFt. George Meade, MD 1-31-88Type Agreement: MDA 904-85-K-005 9-30-87Award Period: From 9/3/85 To 9/3/86 (Performance) 10/20/86 (Reports)Sponsor Amount: This Change Total to Date

Estimated: \$ _____ \$ _____

Funded: \$ 53,199 * \$ 53,199Cost Sharing Amount: \$ N/A Cost Sharing No: N/ATitle: Research in Digital Speech ProcessingADMINISTRATIVE DATAOCA Contact Ralph Grede X 4820

1) Sponsor Technical Contact:

2) Sponsor Admin/Contractual Matters:

Mr. Richard Dean, R 556Mr. Allan J. WitchieMaryland Procurement OfficeMaryland Procurement Office9800 Savage Road9800 Savage RoadFt. Geo. C. Meade, MD 20755-6000Ft. Geo. C. Meade, MD 20755-6000Defense Priority Rating: None Indicated Military Security Classification: None indicated(or) Company/Industrial Proprietary: N/ARESTRICTIONSSee Attached Government Supplemental Information Sheet for Additional Requirements.

Travel: Foreign travel must have prior approval — Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of \$500 or 125% of approved proposal budget category.

Equipment: Title vests with Sponsor - However, no equipment is proposed.COMMENTS:

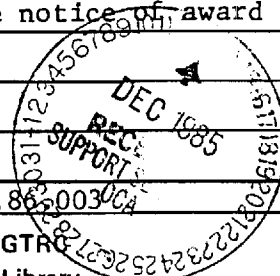
* Total from initial award and modification P00001. The modification and award dates do not coincide. This is acceptable since the notice of award letter was dated 9/30/85.

COPIES TO:SPONSOR'S I. D. NO. 02,123,001.8620030CA

Project Director
Research Administrative Network
Research Property Management
Accounting

Procurement/GTRI Supply Services
Research Security Services
Reports Coordinator (OCA)
Research Communications (2)

GTRC
Library
Project File
Other A. Jones



Georgia Institute of Technology

College of Engineering
School of Electrical Engineering
Digital Signal Processing Laboratory
Atlanta, Georgia 30332

GEORGIA TECH 1885-1985

DESIGNING TOMORROW TODAY

April 7, 1986

Mr. Frank Malkin
U. S. Army Human Engineering
Laboratory
Aberdeen Proving Ground, MD 21005-5001

Dear Frank:

Enclosed, please find: a brief outline of what we have been doing this quarter, and a short write-up resulting from a literature search and analysis performed by us.

We are interested in making field recordings in various non-laboratory environments and need to know a little more information. Basically, do helicopter pilots ever wear masks? My guess is not, but I need confirmation. Also, what is the general set-up for voice communications? Is there a close-talking microphone, throat accelerometer or what? Is it possible to get a sample of speech and background noise from a pilot in flight?

We plan to build up our data base starting in May, and need to have all issues resolved prior to this.

If this report is lacking in some way, please contact me at once. I'm looking forward to meeting you at Speech Tech '86.

Respectfully,

Mark A. Clements

MAC:svs

Enclosures

Progress Report Project R6078-0A0

Automatic Recognition of Speech in Stressful Environments

Principal Investigator: M. A. Clements

During the quarter January 1, 1986 through April 1, 1986, we performed the following:

- 1) Mr. John H. Hansen, Ph.D. candidate, is now working full-time on this project. He and I have written software for enhancement, recognition, and general speech analysis.
- 2) Mr. Hansen and I have performed a detailed literature search and analysis thereof. One of the decisions we have made regards our usage of the word "stress" to mean any perturbation which renders the speech altered. Such would include emotional duress, shouting, rapid speech, and many others. We have commenced to classify different stresses and fully characterize the acoustic-phonetics involved.
- 3) We have performed a search for available data-bases world-wide; and although some would come close to meeting a subset of our needs, none of them are actually suitable. We have made arrangements with two psychiatrists at Emory University (Drs. Philip Ninan and Bernard Holland) to make tape recordings of patients who are known, periodically, to go through periods of fear and anger. Such a corpus would enable capture of these emotions in a non-simulated manner, along with a large amount of baseline data.
- 4) The search for a second graduate assistant has begun. Although many are willing, we are exercising great care. We are, in fact, demanding the willingness to work on the project for one quarter for course credit rather than for support. One such individual, will be starting in this capacity spring quarter if he has passed his preliminary written doctoral qualifying exam (results will be available this week).

INTRODUCTION

Many factors contribute to the success or failure of speech enhancement algorithms. Several approaches have been taken, each attempting to capitalize on various characteristics or constraints of the speech and/or noise signals. A particular techniques' advantages or disadvantages are closely related to the underlying assumptions made in seeking a solution to the corrupted speech problem. One application for enhancement is in the speech recognition area. Thus far, approaches such as dynamic time warping or hidden Markov modeling have largely been applied in tranquil environments. Studies have shown that recognition accuracy is severely reduced when speech is uttered in a noisy, stressful environment. Therefore the necessity exists for development of robust enhancement preprocessors which produce speech less sensitive to varying factors such as stress or background noise. It has already been demonstrated that existing enhancement preprocessors are beneficial in improving intelligibility for bandwidth compression systems. So, it is encouraging that such preprocessors may also improve recognition accuracy for enhancement/recognition systems. Before such algorithms can be developed, it is necessary to understand how factors such as stress and noise effect speech parameters. Section 1.0 therefore addresses the issues of objectively measuring stress and emotion in speech. Enhancement of speech for human listeners requires improvement in quality and/or intelligibility, (and possibly other factors as well). For the purposes of speech recognition by a system however, it is not entirely clear whether improvement in both areas is necessary. This may be dependent on the specific recognition requirements. Section 2.0 compares several enhancement techniques in terms of quality/intelligibility improvement. Direct comparison is sometimes difficult since each class of techniques vary with underlying assumptions in speech and noise characteristics.

1.0 STRESS and EMOTION in SPEECH

I ... uh ... can't even talk to people. It's a, it's a ... I, a, I can't talk, ladies and gentlemen ... I, I can hardly breathe. I, I'm going to step inside where I cannot see it. I, I can't a ... Listen folks, I, I'm going to have to stop for a minute because I've lost my voice. This is the worst thing I've ever witnessed.

(Radio announcer describing the crash of the *Hindenburg*.)

When a person is exposed to an emotion-producing situation, such as the radio announcer describing the approach of the *Hindenburg* zeppelin when it suddenly burst into flames, various changes in bodily state occur. These changes, which are a result of changes in the activity of the sympathetic and parasympathetic branches of the nervous system, can affect speech behavior, even against an individual's will. This section focuses on identifying objectively measurable quantities of stress and emotion in speech. To begin with, situations where stress evaluation is used will be summarized. Next, subjective observations of stress in speech will be examined, followed by a brief summary of how prosodic features in speech are conveyed. Finally, a summary of research findings concerning stress and emotion in speech is presented.

There are many situations where it is desirable to evaluate or monitor the emotional state of a speaker. Three distinct areas where research has been undertaken include; (1) the area of crime countermeasures (e.g., lie detector systems, analysis of phone threats including suicides, assassination or bomb threats, etc. [6,7]), (2) safety and security (e.g., air traffic controllers and pilots in noisy, high stress environments [8,24], deep sea divers [2], astronauts [23], power system operators [25], etc.), (3) and psychology (e.g., emotional state of patient [3,4]). Note that levels of emotion or stress vary greatly in these examples. In addition, subjects may be uncooperative and attempt to hide their emotional state, (i.e. lie detector subjects). An area of speech processing where emotional content is important is speech recognition. For example, speech synthesis/recognition in an aircraft cockpit has been considered as a means of reducing task demands on pilots [11,16,17,18,24]. Here, the pilot is exposed to a high ambient noise environment while under a high level of stress requiring constant monitoring of instruments, terrain, and other aircraft. Malkin and Christ [11], placed speakers in a 107 dBA helicopter noise environment. Speakers were required to press response buttons corresponding to randomly displayed symbols while speaking into a recognition system.

Results showed a decrease of 23% from the ideal 99% recognition rate. In similar experiments, Simpson [24] considered an isolated-word, speaker dependent recognition system. Vocabulary size consisted of six words normally used in a cockpit, and pilot stress involved a computer simulated gunner/scout pursuit task. Final conclusions from these experiments strongly recommended against the use of present day speech recognition in this environment due to significant loss in recognition accuracy. It is quite obvious that the mental loading simulated in these experiments is far less taxing than that observed during a critical emergency situation. Therefore, more robust speech processing algorithms, insensitive to speaker stress, are needed if voice I/O is to be used in such environments. Before such algorithms can be developed, reliable objective indicators of stress in speech must first be established.

1.1 Stress and Prosodic Features in Speech

Stress is a psychological state that is a response to a perceived threat and is accompanied by specific emotions (e.g., fear, anxiety, anger). Verbal markers of stress range from highly visible to invisible markers as perceived by the listener. It is important to note that if a marker is visible to a listener, it is also visible to the speaker. Therefore, along with a continuum of visibility, there is a continuum of speaker control. If a speaker wishes to hide his emotional state, high visible markers would be the first to change, since they possess the greatest level of speaker control. If a speaker is not concerned with visibility of emotional state, then analysis of the most visible markers would possibly be a more reliable indicator of stress. Four classes of markers exist: visible, near-visible, less-visible, and invisible. Visible markers refer to verbal quantity and speaking rate. For example, when a speaker is experiencing fear, he wishes to transmit as much information as quickly as possible. Near-visible markers include speech faults such as sentence change, stutter, tongue-slip, etc. Figure 1 summarizes the most common of these. Less-visible markers include filled and unfilled pauses. These represent periods between utterances, and function as a buffer period where the speaker organizes his thoughts. An unfilled pause represents a greater amount of speaker control for the class of less-visible markers. The filled pause is characteristic of the "uh" sound during sentence construction. The filled pause manifests itself when a speaker attempts

to plan the next verbal sequence too quickly. The speaker is usually unaware of this trait (low amount of speaker control). However, it may not always indicate stress, since some people say "uh" all the time. The last class of speech markers are the invisible, which possess the least amount of speaker control. Invisible markers refer to lexical leakage which is defined as the choice of words, influenced by unconscious and preconscious background factors. For example, a farmer applying for drought relief complains, "We're drowning in red tape." Here, the word "drowning" is inspired partly by the need for water. Examples of visible, near-visible, and less-visible markers can be seen in the *Hindenburg* announcer's message. Research in psychiatry conclude that verbal markers are not entirely successful or consistent in predicting/analyzing stress [4]. Many markers are continuously monitored both consciously and subconsciously by the speaker, and thus are prone to correction. Therefore, other objective variables must be considered in determining stress. At this point, it may be beneficial to consider physiological aspects of speech production, specifically prosodic features. Understanding how prosodic features convey linguistic information will be useful in the analysis of stress and emotion in speech.

The rhythms and intonations of language are formed using prosodic rules. These features usually extend over more than one phoneme segment and are therefore called suprasegmental. Prosodic features which carry linguistic information include stress and intonation. Stress in this context refers to lexical stress where a given syllable in an utterance is further emphasized. (For example, compare the two phrases *That's just in sight*, and *That's just insight*. The stressed and unstressed syllables "in" and "sight" are reversed in each sentence giving an entirely different meaning.) Prosodic features are produced by changes in the glottal sound source and the timing of articulatory movements. The glottal source factors operate through actions of the speech breathing muscles and the vocal folds; the timing factors operate through the movements of the upper articulators. The two main factors in the glottal source variation for prosodic features include subglottal air pressure and tension of the vocal folds. These affect the fundamental frequency, the amplitude, and the source spectrum. Increasing force on the lungs causes an increase in subglottal air pressure; which results in an increased

1. Sentence change	Well she's . . . already she's lonesome.
2. Repetition	Cause they . . . they get along pretty well together.
3. Stutter	It sort of well I . . . I . . . leaves a memory.
4. Omission	She mour . . . was in mourning for about two years.
5. Sentence incompletion	Well I'm sorry I couldn't get here last week so I could . . . ah . . . I was getting a child ready for camp and finishing up swimming lessons.
6. Tongue-slip	We spleat the bitches (for "split the beaches").
7. Intruding incoherent sound	I see a girl now I'd like to take out . . . I just . . . dh . . . ask her.

Figure 1: Examples of Near-Visible Markers. (Goldberger and Breznitz [4])

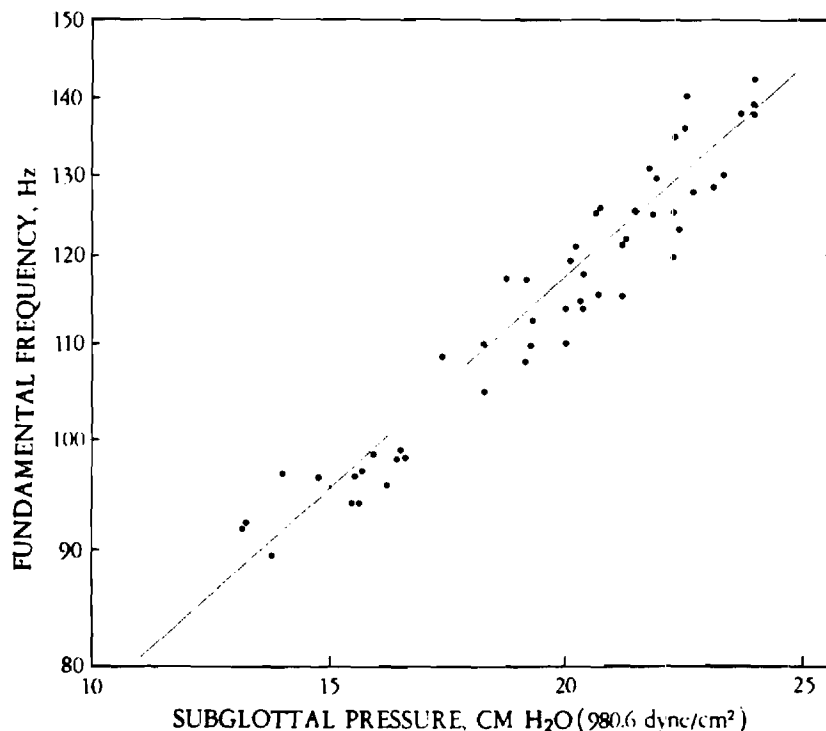


Figure 2: Relationship between fundamental frequency and subglottal air pressure. The group of points between 90 and 100 Hz are for the speaker's normal phonations, other points resulted from applied chest pressure. (Frequency plotted on log scale, Pickett [14])

rate of airflow pulses emitted by the glottis. This increased rate corresponds to an increased fundamental frequency (f_0) and amplitude. The relationship between subglottal pressure and voice pitch for a vowel sound is shown in figure 2. Another problem concerns how various stress patterns are produced by changes in subglottal air pressure and muscle tension on the vocal folds. These patterns depend on location of lexical stress (first or second syllable in a word), and type of utterance (statement or question). Consider the variations in subglottal air pressure and f_0 with time in the four sentences of figure 3. The intonation contour for the statement generally shows a downward movement of pitch (f_0) from beginning to end, which may be correlated with a decrease in subglottal pressure toward the end of the statement. However, superimposed on this may be variations in the pitch due to syllable stress of the utterance. For the question, the pitch contour rises during the utterance because of increased vocal fold tension and, again the stressed syllable has a higher pitch than the other syllables. The stressed syllable generally corresponds to a peak in the subglottal pressure to raise the syllable pitch. The breath group theory of intonation, formulated by Lieberman [9], suggests how basic patterns of subglottal pressure and vocal fold tension are encoded to produce the effects of stress and intonation. The theory simply states that the intonation contour will naturally always fall toward the end of a breath group because of the lower subglottal pressure just before taking a new breath of air (see fig. 4). Therefore, a basic breath group contour can be marked as a question through an increase in vocal fold tension. This theory has received criticism due to its simplicity in assuming independence of vocal fold tension and subglottal pressure in determining pitch. Recent studies conclude that intonation contours are controlled primarily by vocal fold tension in the high-pitch parts of the contour and subglottal pressure in the lower pitched parts. Also, the vertical position of the larynx is believed to be the adjustment factor in determining which control factor dominates.

Other glottal source factors related to stress and intonation are sound intensity and spectral balance between low and high regions of the glottal source spectrum. An increase in vocal effort increases subglottal pressure, and if vocal fold adjustments are maintained, the resulting glottal flow pulse will be steeper with sharper corners. This raises the high

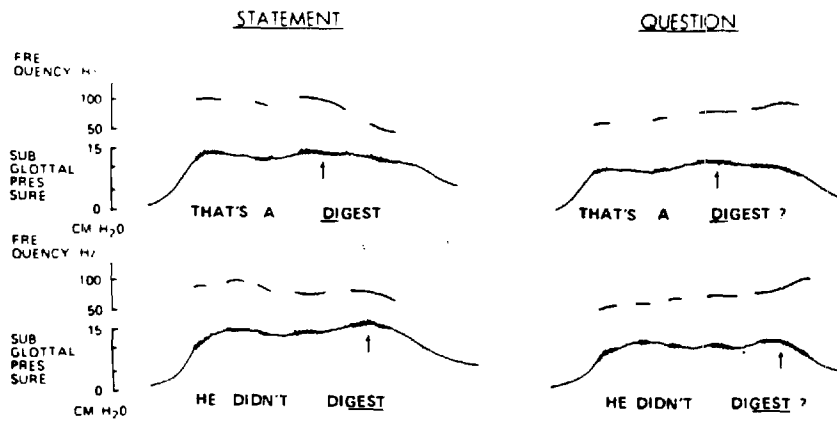


Figure 3: Relations between fundamental frequency contours and subglottal air pressure for statements and questions with two different patterns of word stress. The stressed syllable is underlined and indicated by an arrow under the contour of subglottal pressure. The intonation contours are above the contours of subglottal pressure. It can be seen that the stressed syllables correspond to peaks in the pressure contours and that the intonation contours rise for the questions and tend to fall for the statements. (Pickett [14])

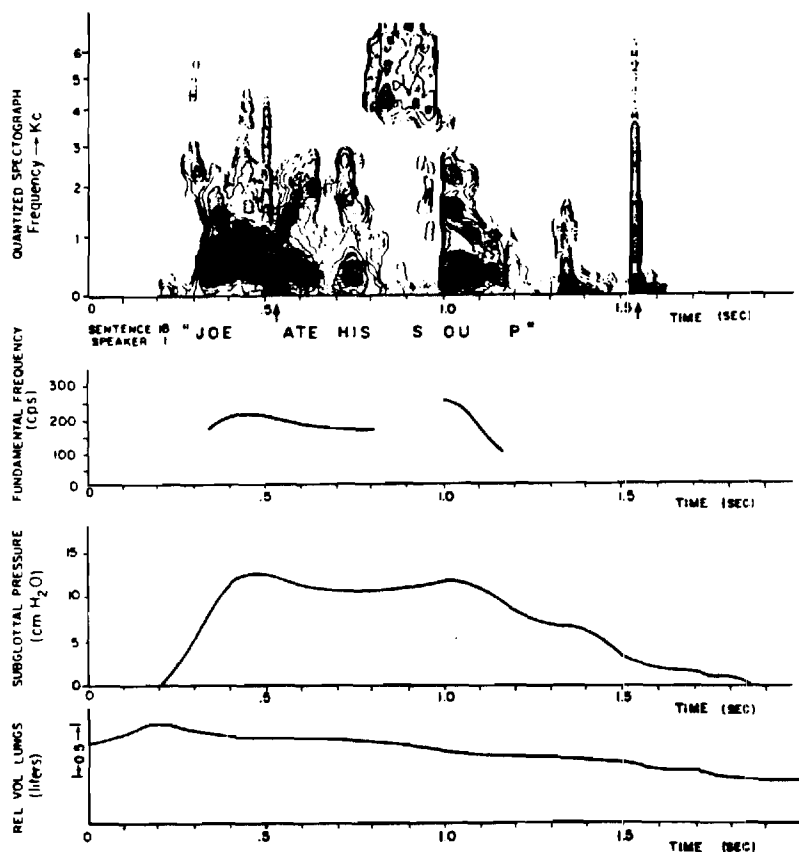


Figure 4: Acoustic and physiologic data for a speaker reading the sentence *Joe ate his soup*. A normal unmarked breath group was used. The fundamental frequency, subglottal air pressure, and the volume of air in the speaker's lungs are plotted as functions of time. A quantized sound spectrogram is also presented. (Lieberman [9])

frequency portion of the source spectrum relative to the low frequency part. Increased vocal fold tension can also affect spectral balance by boosting the high frequency end of the glottal spectrum. Thus, stressed vowels are found to have higher amplitudes for all formants (except F1) than unstressed vowels [9,14]. A final observation concerns durational variations that arise from prosodic conditions. Many rules have been formulated by phoneticians for synthesizing correct speech. One important rule is that syllable stress causes a lengthening of the vowel. Other rules deal with various vowel/consonant durational patterns. A summary of the major sound-source factors in prosodic features are:

- (1) f_0 for vowels is higher in stressed syllables than in unstressed syllables, (due to subglottal pressure and vocal fold tension). Also, voice pitch shows a contour variation over syllables that corresponds to the grammatical function and structure of the phrase.
- (2) The intensity of vowels in stressed syllables is higher than in unstressed syllables.
- (3) Spectral balance of vowels is affected by the higher vocal effort on stressed syllables, resulting in relatively more intensity of F2 and higher formants.

Analysis of prosodic features including intonation and lexical stress have been investigated for isolated-word recognition by Aull and Zue [1], and also for the purposes of producing more natural sounding speech in synthesis systems by Nakatani and Aston [12]. It is believed that knowledge of prosodic features and how they effect fundamental frequency, energy, and duration, will help explain the processes inherent in how humans encode stress and emotion in speech. At this point, problems of objectively measuring stress in speech can now be addressed.

1.2 Acoustic Correlates of Stress/Emotion in Speech

Due to lack of research, characteristics of speech produced under stress remains sketchy at best. Thus far, research has been limited in scope, often using only one or two subjects and analyzing a single parameter (usually f_0). It is not unusual for researchers to report conflicting results, due to differences in experimental design, level of actual or simulated stress, or interpretation of results. For example, some studies concentrate on analysis of recordings from actual stressful situations [8,23,25,26]. There is usually little doubt as to the presence of stress in these recordings, however a quantitative analysis can only be carried out if recordings of the talker speaking the same utterances under stress-free conditions is available. In addition, some researchers argue that speakers in these situations may experience

several emotions simultaneously, (e.g., *Hindenburg* announcer most likely experienced fear, grief, and anxiety). Another group of studies have been performed using simulated stress or emotions [5,6,7,10,26]. This work offers the advantage of a controlled environment, where a single emotion can be examined with little background noise. In some cases, variable task levels of stress have been used. Other advantages include larger data sets with multiple speakers. This allows results to be based on general speaker characteristics instead of possibly particular characteristics of an individual speaker in conveying emotion. The disadvantages in these studies have been the reduced task level. There are obvious questions concerning the ethics of inducing tremendous stress levels upon subjects. In addition, studies using actors may produce exaggerated caricatures of emotions in speech.

Analysis of prosodic features revealed varying f_0 as a key factor in transmitting linguistic information such as lexical stress and intonation. Other factors included intensity, spectral balance, and durational effects. These parameters may represent possible carriers of emotion or situational stress. In previous work [3], Williams, Stevens, and Hecker found the acoustic properties that appeared to be the most sensitive indicators of emotion were attributes that specified the contour of f_0 throughout an utterance. There are several reasons why changes in f_0 with time provides information on emotional state. First, there is considerable latitude in variations of f_0 , since only certain aspects of the f_0 contour carry information with regard to linguistic content of a message. The principal linguistic functions of f_0 changes are to indicate lexical stress, and mark boundaries of breath groups. With only these constraints, a speaker is relatively free to use changes in f_0 to convey nonlinguistic information such as emotion. Second, the preoccupation with f_0 contours comes from the literature on physiological correlates of stress. For example, respiration is frequently a sensitive indicator in certain emotional situations: when an individual experiences a stressful situation, his respiration rate increases. This presumably will increase subglottal pressure during speech. As was seen in figure 2, increased subglottal pressure resulted in a corresponding increase in f_0 during voiced sections. An increased respiration rate also leads to shorter durations of speech between breaths which would effect the temporal pattern

(articulation rate). The dryness of the mouth found during situations of excitement, fear, anger, etc., can also effect speech production (e.g., muscle activity of larynx and condition of vocal cords). Muscle activity of the larynx and vibrating vocal cords directly effect the volume velocity of air through the glottis, which in turn affects f_0 . Other muscles (tongue, lips, jaw, etc) shape the resonant cavities for sound and therefore do not have a direct influence on f_0 . Any analysis of the speech signal that reflects vocal cord activity is more likely to be influenced by physiological changes brought about by the emotional state of the speaker. These physiological changes suggest, that as subglottal pressure increases, the resulting individual glottal pulses become narrower. Therefore, a change results in the spectrum of the pulses. Qualitative changes in glottal pulses can be obtained by observing wide-band spectrograms.

Studies carried out by Williams and Stevens [26] reflect a significant contribution to the field. Two avenues for analysis were considered: first, a detailed analysis of "field" recordings in actual emotional environments, (recordings of the radio announcer during the *Hindenburg* disaster), and second, analysis of recordings of "method" actors simulating fear, anger, sorrow, and neutral emotions. A unique approach had the method actors simulate the same real-life situation. This was done to validate the use of actors in simulating emotions. For the simulated conditions, actors performed a play where selected phrases (control clusters) were spoken by the same actor in different emotional situations. Control clusters could therefore be carefully analyzed with emotion being the only distinguishing variable. Analysis of simulated and actual emotional situations from several research studies will be presented separately.

1.21 Analysis Using Simulated Stress or Emotions

Analysis of studies using simulated stress/emotion will be considered first. Williams and Stevens [26] used method actors to simulate emotions. They considered six areas for parametric analysis: fundamental frequency contours, f_0 variability, characteristics of wide and narrow-band spectrograms, mean rates of articulation, and finally, average spectral content. Hicks and Hollien [6,7] simulated stress by using mild electrical shock. Their results measured

fundamental frequency, speech intensity, and speech rate. Hecker et al. [5] simulated stress by having subjects perform a timed arithmetic task. Parameters included fundamental frequency, speech level, and spectrogram characteristics. Examination of each of the areas suggested by Williams and Stevens as well as comparisons with Hicks and Hollien, and Hecker et al. follow.

Fundamental frequency f_0 contours were analyzed for neutral, anger, sorrow, and fear (see fig. 5). For the neutral condition, changes in f_0 were relatively slow and the shape of the contour throughout the utterance was smooth and continuous. For anger, f_0 was generally higher throughout the utterances with one or two syllables characterized by large peaks in f_0 . For fear, the f_0 contour departed greatly from the neutral situation. Hicks and Hollien found similar increases in f_0 . However, Hecker et al. observed conflicting results. Some subjects increased, while others decreased f_0 .

Variability of f_0 was observed from narrow-band speech spectrograms (see fig. 6). Ranges for sorrow and neutral emotions were similar. Anger had the highest mean f_0 and the widest f_0 range on a linear scale. For fear, the variability range was skewed with some very high values of f_0 . Also, fear showed a wider and higher range of f_0 than neutral. Hicks and Hollien found similar results for stress (fear and anger). Hecker did not consider variability.

Wide-Band spectrograms reflected changes in utterance duration and vocal-cord vibrations. The duration was the shortest for neutral, longer for both fear and anger, and the longest for sorrow. Increases were due in part to increased vowel duration, but primarily from lengthened intervals of closure/vocal tract constriction for the consonants. Hicks and Hollien supported these findings. Hecker et al. observed several effects from wide-band spectrograms but they varied considerably between subjects. One effect occurring in several subjects was a lengthened and irregular glottal period at the end of the breath group. Another observation was the amount of high frequency energy in the glottal pulses (see fig. 7). This caused the third and fourth formants to become generally weaker for the stressed versus neutral condition.

Analysis of Narrow-Band spectrograms confirmed measurements of f_0 (see fig. 8). For anger, the highest average f_0 was observed which possessed the most rapid frequency changes. For fear, rapid fluctuation/tremor was observed in f_0 . Also, in some cases f_0 would start high at the

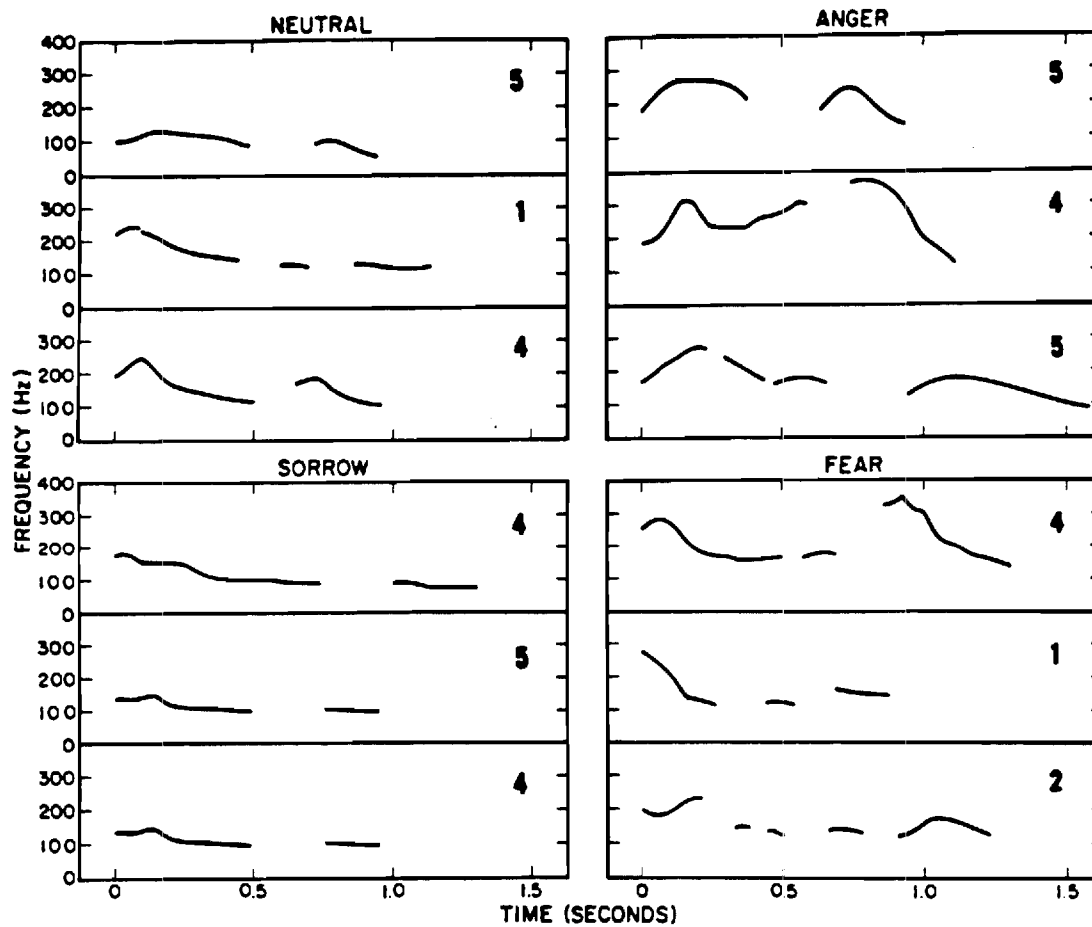


Figure 5: Fundamental frequency (f_0) contours of selected control clusters spoken by a "method" actor during various situations in the scenario. Numbers refer to control clusters.
(Williams and Stevens [26])

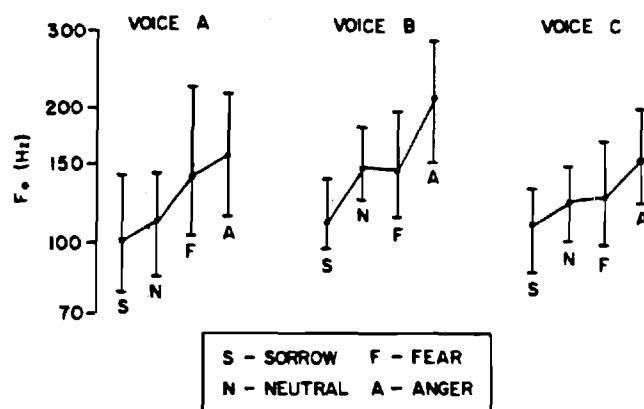


Figure 6: Median f_0 and range of f_0 for each of three voices speaking in neutral, sorrow, fear, and anger situations. (Williams and Stevens [26])

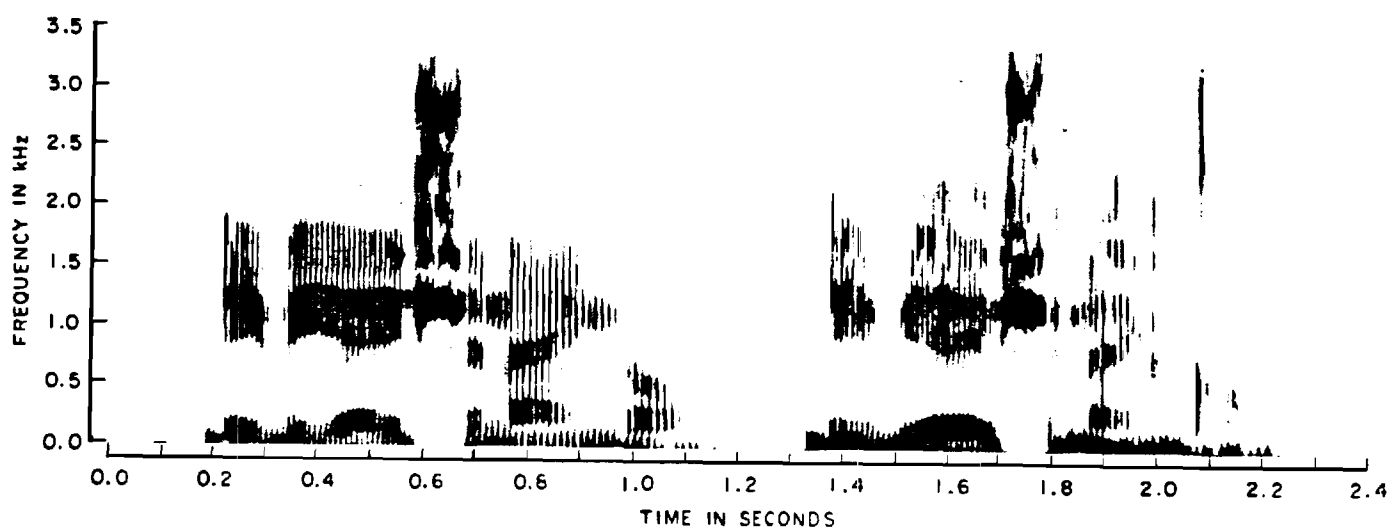


Figure 7: Spectrograms of the test phrase *Relative Velocity* spoken in control condition (left) and under stress (right). Note the very long intervals between glottal pulses and some irregularities in the pulses near the end of the utterance produced under stress. Change in high-frequency energy in the glottal pulses (seen more in front vowels than back vowels due to more high frequency energy in front vowels) causes the third and fourth formants to become generally weaker for the stress condition than control. {Compare F3,F4 in 0.6-0.8 with 1.7-1.9} (Hecker et. al., [5])

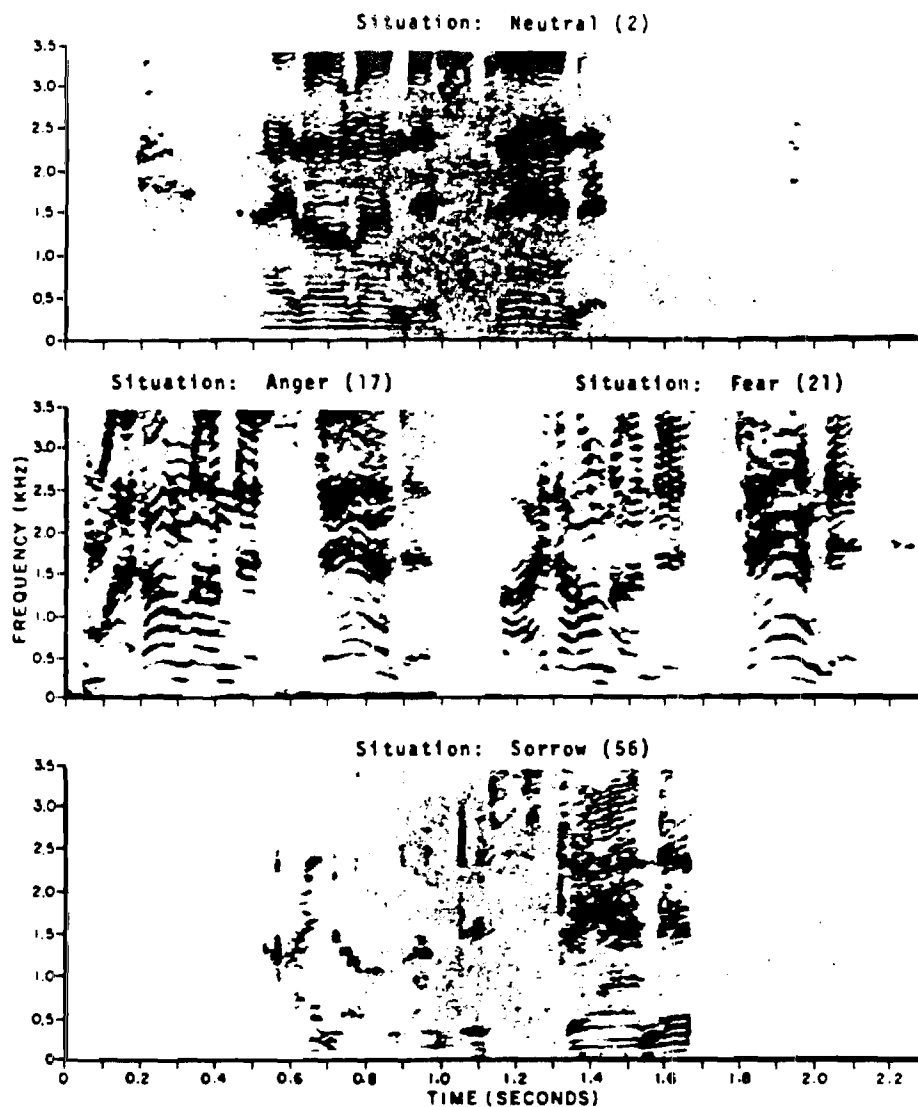


Figure 8: Narrow-band spectrograms of the control cluster *I don't understand it* spoken in four situations. (Williams and Stevens [26])

	Neutral	Anger	Fear	Sorrow
Voice A	4.03	4.26	3.92	1.84
Voice B	4.89	4.32	3.90	2.03
Voice C	4.02	3.88	3.57	1.86
<i>Mean</i>	4.31	4.15	3.80	1.91

Figure 9: Mean rate of articulation (syllables per second) for each of the three voices speaking in different situations. (Williams and Stevens [26])

beginning of a syllable and then fall with an initial bump. Hecker et al. observed less fluctuations in f_0 under stressed conditions.

The **mean rate of articulation** in syllables/second was found for each of the emotions (see fig. 9). The order from fastest to slowest were neutral, anger, fear, and sorrow. Hicks and Hollien observed similar results.

The **average spectral content** was found for the three simulated emotions. Anger and fear both possessed low amounts of energy in the low frequency band. The variation was not consistent for sorrow or neutral conditions. For the high frequency bands, anger and fear possessed larger amounts of energy while sorrow always had the least.

One parameter not considered by Williams and Stevens was **speech intensity** during voiced sections. Hicks and Hollien, Hecker et al. all measured intensity levels. Both found inconsistent results from one test phrase to the next. However, the simulated stress conditions in these experiments may not have been adequate for generalizations to other environments. Pisoni et al. [15] investigated acoustic-phonetic correlates of speech produced in noise. In this study, subjects spoke in quiet and in 90 dB SPL white masking noise environments. The masking noise was supplied through headphones causing only the speaker to experience the background noise with the recorded speech noise free. Results showed an increase in overall amplitude of vocalic sections, increased duration, increased average f_0 , and a spectral tilt where upper formants became more intense in the noise condition. Lieberman and Michaels [10] instructed subjects to simulate eight emotional states. Their creative approach was to select a parameter as an emotion relay, extract that parameter from the speech, and observe whether the resulting sound could correctly be identified as the simulated emotion by listener groups. Figure 10 shows that fear (or stress) was highly identified using only amplitude information with constant pitch. Another body of work which may be relevant to speech intensity include studies on shouted speech. Rostolland [19,20] performed acoustic and phonetic studies and observed reduced intelligibility in shouted speech. This in part, was due to increased energy in the higher frequency regions resulting in altered spectral shape. Rostolland also observed increased f_0 with reduced variability. This may account for conflicting results in f_0

variability (i.e., speech shouted in some studies, while simply spoken under stress in others). It seems intuitive that an individual experiencing fear (e.g., panic such as a life threatening situation), would tend to raise his voice.

1.22 Analysis Using Actual Stress or Emotional Situations

A comparison of results from actual stress situational recordings is somewhat difficult, due to varying parameters measured and levels of stress experienced. However, they are important since their analysis may help verify experimental procedures and results from simulated studies.

Kuroda et al. [8] analyzed tape recordings of 14 pilots with varying mission experience in actual aircraft accidents, 8 of which were ultimately fatal. Their analysis consisted of finding the vibration space shift rate (VSSR) in speech spectrograms.† The VSSR was classified into nine bands corresponding to varying relative pitch periods. Figure 11 illustrates a typical case example of a pilots shift rate profile. The corresponding phrase used to find the VSSR is indicated for each phase of the mission. The ultimate conclusions were that as stress increased, so did f_0 .

Simonov and Frolov [23] analyzed recorded communications of a cosmonaut at various flight stages. Analysis consisted of monitoring heart rate and the spectral centroid of the first vocal tract formant. Some correlation was found, but the conclusions were that further research was necessary.

Streeter et al. [25] carried out a more complete analysis of a telephone conversation between a system operator (SO) and his superior chief (CSO) working at the Con. Edison power plant prior to the 1977 New York blackout. Analysis consisted of pitch statistics, and amplitude and timing measurements. An attractive feature of the data was the increased situational stress throughout the hour long conversation. Results were somewhat conflicting since it appeared SO was passing decision making authority over to CSO during the emergency. Streeter et al., discovered no reliable and valid acoustic indicators of psychological stress. It should be noted that results for CSO showed increased pitch and amplitude over time, which

† VSSR is related to differences in pitch period between normal and stress situations.

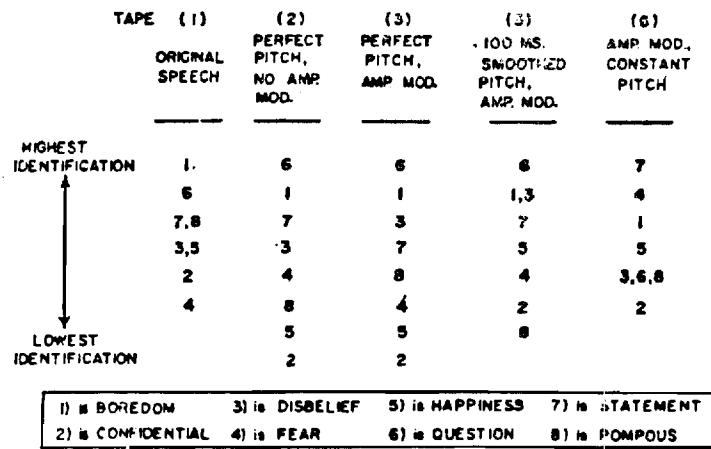
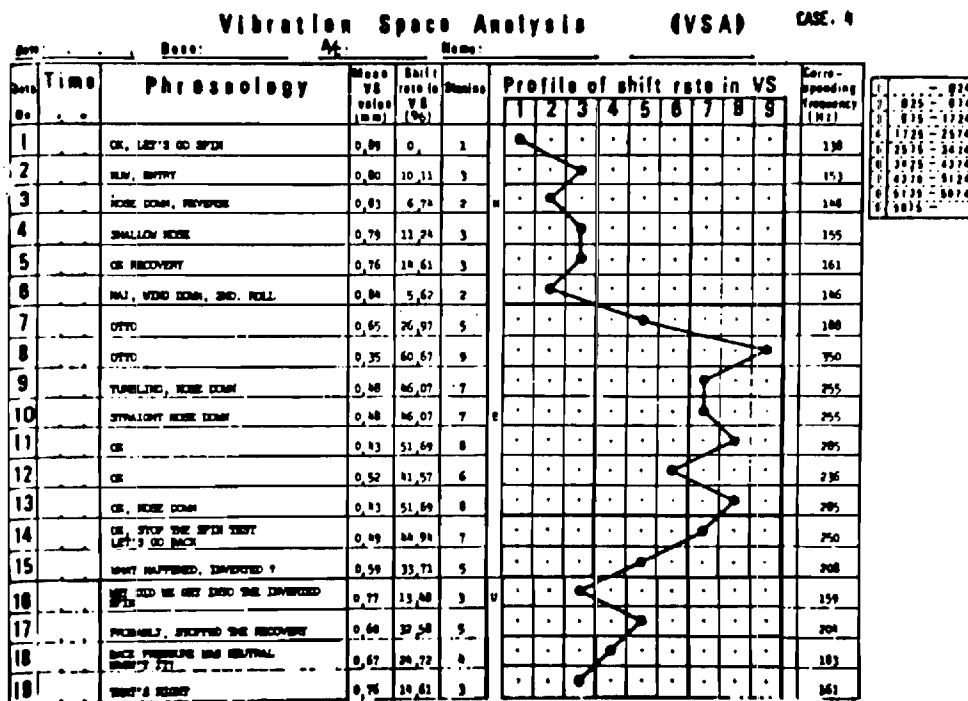


Figure 10: Relative listener identifiability of an emotional state for each processing condition. (Lieberman and Michaels [10])



Case 4. A 32-year-old T-33A test pilot with 2,298 h of flying time had an uneventful takeoff prior to attempting spin recovery maneuvers. He inadvertently entered into an inverted spin at the recovery spin test. Since ejection was impossible due to the low altitude, both pilots concentrated on the recovery from the spin. The extreme tension (phrase #7 and 8) is denoted by the shouts recorded. After an adequate recovery, the VSSR gradually declined phrase #14)

Figure 11: Typical shift rate profile for a pilot during an emergency situation. (Kuroda et. al., [8])

agree with simulated stress studies. One conclusion Streeter et al. noted was that while speaker behavior appeared unpredictable, listener behavior was predictable. Results showed that listeners' referred to a vocal stereotype which they associate with stress which includes elevated pitch and amplitude, as well as increased variability in these parameters.

Finally, Williams and Stevens [26] performed analysis of the recorded radio announcer during the *Hindenburg* disaster. In an effort to justify results from their simulated emotions, they had their "method" actors recreate the announcer's message. Narrow-Band spectrograms (see fig. 12) were compared for the announcer and actor. Results were not entirely consistent, though increased average f_0 along with tremor (irregular bumps in contour) were observed for both. Median f_0 and f_0 range were also found for both (see fig. 13). Increased median f_0 and range of f_0 was observed for announcer and actor, with larger variations for the actor. This would indicate that the actors' emotional expressions may have been over emphasized.

1.3 Summary of Stress and Emotion Research in Speech

The aspect of the speech signal that appears to provide the clearest indication of emotion or stress of a talker is the contour of f_0 versus time. Other parameters considered included f_0 variability, duration, rate of articulation, speech intensity, and characteristics of the vocal tract spectrum. There seems to be no singly reliable acoustic indicators of psychological stress or emotion. This is caused by the varying techniques speakers use to convey their emotional state. Some speakers tend to raise a given parameter, while others decrease it. If a speaker is attempting to hide their emotional state, highly visible markers/parameters are usually corrected first, leaving only subtle and unreliable markers/parameters left for analysis. It may not be possible to develop a quantitative automatic procedure for emotion evaluation, however general trends are known for median f_0 and range of f_0 (Δf_0). These can be summarized for the emotions considered: sorrow results in decreased f_0 , Δf_0 ; anger and fear (situational stress) result in increased f_0 , Δf_0 . These variations assume that the neutral f_0 , Δf_0 are known.

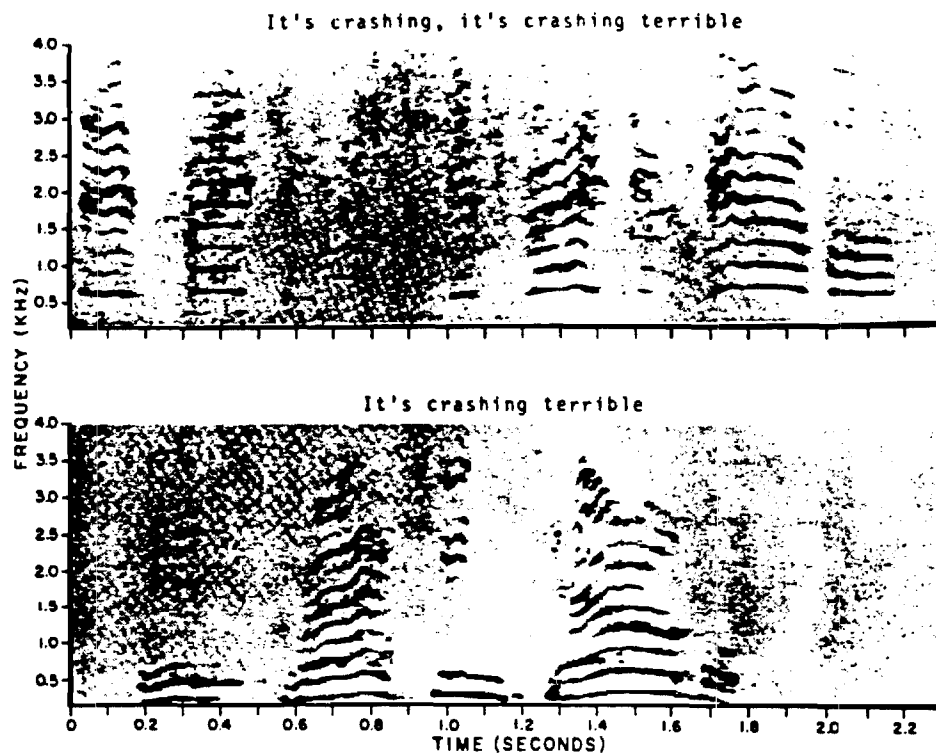


Figure 12: Radio announcer (top) speaking during the crash of the *Hindenburg* and a "method" actors simulation of the announcement. (Williams and Stevens [26])

	F_0	F_0 Range
<i>Original Radio Description</i>		
Before crash	166	124-196
After crash	196	152-260
<i>Voice C's Simulated Description</i>		
Before crash	138	117-168
After crash	222	117-280

Figure 13: Median fundamental frequency (f_0) and range of fundamental frequency for selected utterances obtained from the original radio description of the *Hindenburg* disaster and from a "method" actor's simulated announcement of the same event. (Williams and Stevens [26])

In terms of future work, several researchers have taken steps in attempting to provide a more complete investigation to the problem. Russell and Moore [21] have developed an extensive database for research in this area. Peckham [13] has developed a hardware pitch tracking device which carries out some statistical analysis for determining speaker stress levels. And finally, Scherer and Ladd [22] have an approach which attempts to remove all speech characteristics, leaving only those parameters which convey emotion information. This approach is somewhat difficult since parameters tend to be interrelated, and results are based on listening groups. Further studies in developing objectively measurable parameters of emotional state could help in developing a more complete and natural sounding model for speech processing.

2.0 SPEECH ENHANCEMENT SYSTEMS: A Comparative Analysis

The successfulness of an enhancement algorithm rests on the goals and assumptions used in deriving the approach. Depending on the specific application, a system may be directed at one or more objectives such as improving overall quality, increasing intelligibility, reducing listener fatigue, etc. Several issues and constraints should be noted before an evaluation is attempted. Three assumptions normally made include: i) noise distortion is additive, ii) only the degraded speech signal is available, iii) noise and speech signals are uncorrelated. Constraints placed on the speech model improve the potential for separating speech from background noise. However, such a system would also be more sensitive to "deviations" from these constraints. The same holds for noise assumptions. Confining the noise type, improves the chances of removing it, however, at the expense of dedicating the technique to a specific distortion, such as wide-band random, competing-speaker, or impulsive. Three classes of enhancement systems will be considered, although performance evaluation is difficult due to the fact that appropriate criteria are heavily dependent on specific applications, (e.g., relative importance of quality, intelligibility, listener fatigue, etc. vary between applications). Therefore, results from each class will be presented separately. Continuing from section 2, it would be desirable to compare perceived quality and subjective intelligibility for each class. Unfortunately, such studies have not been undertaken for each system. However, some comparative

results do exist and will be cited. An in depth derivation of each algorithm is out of the scope of this section, therefore an appendix is included which further describes these systems in detail.

Spectral subtraction is one technique based on direct estimation of the short-time spectral magnitude. Noisy phase information is not an issue since the auditory system is relatively insensitive to phase [47]. In this approach, speech is modeled as a random process to which uncorrelated random noise is added. It is assumed that the noise is short-term stationary, with second-order statistics estimated during silent frames. The estimated noise spectrum is subtracted from the transformed noisy input signal. Questions concerning the subtraction procedure arise since results are not guaranteed positive. Different systems remedy this by either half-wave rectification, full-wave rectification, weighted difference, etc. Boll [29,30] pioneered much of the work in spectral subtraction. A particular investigation considered a modified approach which incorporated magnitude averaging of the noisy spectrum. The noise degradation came from a helicopter environment. Results were presented in terms of intelligibility (DRT scores) and quality (a course measure related to DAM profile) for two applications. Results from DRT and quality ratings for enhancement of helicopter speech (with and without magnitude averaging) indicate that spectral subtraction alone does not decrease intelligibility, but does increase quality, especially in the areas of increased pleasantness and inconspicuousness of noise background (see fig. 14a,b). In addition, Boll considered the application of enhancement as a preprocessor for a bandwidth compression system. Results clearly indicate that spectral subtraction can be used to improve intelligibility and quality of speech processed through an LPC bandwidth compression system (see fig. 14c,d). One shortcoming of this system is the resulting "musical tones" after processing. Peaks and valleys exist in the short-term power spectrum of the noise, and their frequency locations and amplitudes vary from frame to frame. When the smoothed estimate of the noise spectrum is subtracted from the actual noise spectrum, all spectral peaks are shifted down while the valleys are set to zero. Thus, after subtraction, there remain peaks in the noise spectrum. The wider peaks are perceived as time varying broadband noise, the narrower peaks as time varying

a)

DIAGNOSTIC RHYME TEST SCORES

	Original	\hat{S} (No Average)	\hat{S} (Three Average)
Voicing	95	92	91
Realism	82	78	77
Sustention	92	87	86
Sibilant	75	83	84
Graveness	68	70	66
Compactness	88	87	88
Total	84	83	82

b)

QUALITY RATINGS

	Original	\hat{S} (No Average)	\hat{S} (Three Averages)
Naturalness of Signal	63	60	61
Inconspicuousness of Background	36	38	42
Intelligibility	30	32	33
Pleasantness	20	31	25
Overall Acceptability	27	33	29
Composite Acceptability	26	32	29

c)

DIAGNOSTIC RHYME TEST SCORES

	LPC on Original	\hat{S} without averaging	LPC on \hat{S} with averaging
Voicing	84	90	86
Realism	56	63	52
Sustention	49	52	56
Sibilant	61	70	68
Graveness	61	62	59
Compactness	83	83	93
Total	66	70	72

d)

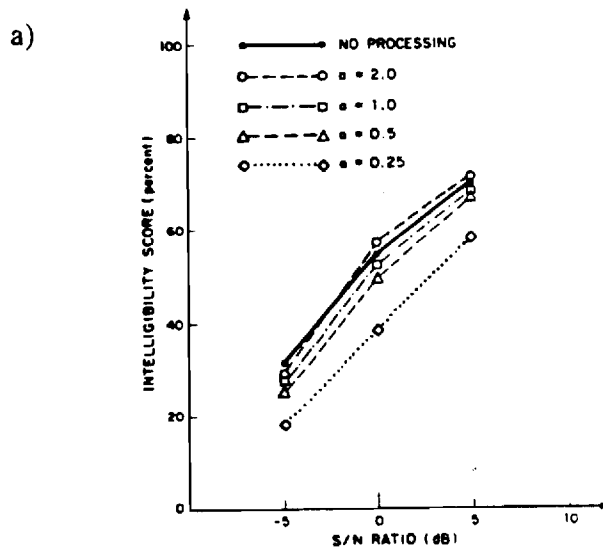
QUALITY RATINGS

	LPC on Original	\hat{S} without averaging	LPC on \hat{S} with averaging
Naturalness of Signal	53	49	58
Inconspicuousness of Background	34	36	39
Intelligibility	28	30	28
Pleasantness	15	28	20
Overall Acceptability	24	28	26
Composite Acceptability	23	29	25

Figure 14: Spectral noise subtraction results from intelligibility tests (DRT) and quality tests (course measure somewhat related to DAM profile). a) and b) are results from speech enhancement in a helicopter noise environment. c) and d) are results from an LPC based enhancement/bandwidth compression system. Results are given with, and without magnitude averaging. (Boll [29,30])

tones. Berouti et al. [28] proposed a method for reducing these tones by overestimating the noise spectrum, thereby subtracting off more of the noise energy. Unfortunately, a comparative analysis of quality/intelligibility was not performed. Peterson and Boll [44] extended the technique by applying spectral subtraction in separate frequency bands, tuned to the loudness components perceived by the auditory system. Hanson and Wong [36] applied this technique to the competing speaker problem. Their approach concluded an increase in intelligibility; heretofore never accomplished. Greatest improvement occurred for low SNR (-12dB) with smaller levels of improvement as SNR increased. However, exact pitch information was used for their evaluation, something very difficult to achieve at such a low SNR.

Correlation subtraction techniques are sometimes referred to as spectral subtraction, since their basic premise is the same. Processing involves subtracting an estimated noise autocorrelation from the noisy speech autocorrelation function. One generalization of correlation subtraction formulated by Weiss et al. [49] involves raising the noisy magnitude spectrum to a power a , prior to subtraction. This system, termed INTEL was evaluated by Lim [37] for wide-band random noise under varying values of a . Figure 15 gives intelligibility scores based on tests involving nonsense sentences. Results show that intelligibility is not improved. By setting $a=1$, the resulting system is equivalent to Boll's spectral subtraction technique. These results appear to contradict each other (Boll concluded no decrease in intelligibility); yet this can be attributed to different noise degradations: helicopter noise for Boll and wide-band random noise for Lim. It was also observed that processed speech with $a=1$ or 0.5 sounded distinctly "less noisy" and of "higher quality" at relatively high SNR. Berouti et al. [28] considered a gain factor k along with the power factor a in the subtraction process. McAulay and Malpass [42] included a gain factor in the subtraction process based on the probability of only noise being present. Their argument was that further spectral reduction is desirable if the probability of only noise is high. Weiss and Aschkenasy [48] later refined their INTEL system to incorporate a second transform operation. This resulted in performing the subtraction process in the cepstral domain. Some improvement in intelligibility was observed, however no quantitative analysis was carried out. (See fig. 15 for various system diagrams.)



b)

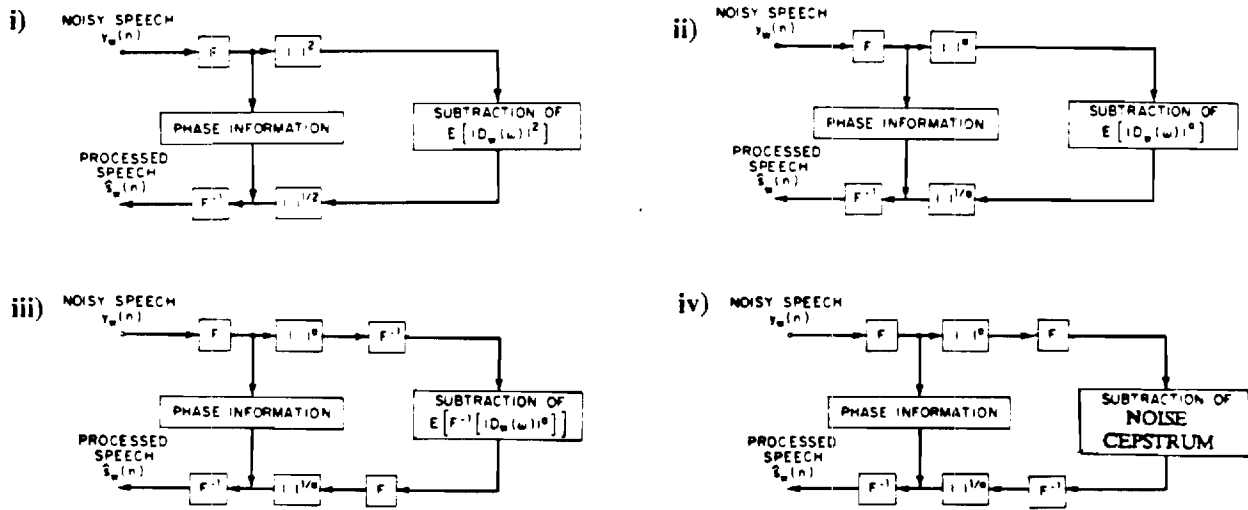


Figure 15: a) Intelligibility scores of a spectral subtraction technique for enhancement of speech degraded by wide-band random noise. (Lim [37])

b) Various implementations of power spectral subtraction.

- Original Spectral Subtraction from Boll [30].
- Generalized Spectral Subtraction from Weiss et. al., [49].
- Generalized Correlation Subtraction.
- Improved INTEL system from Weiss and Aschkenasy [48].

Short-time Wiener filtering is a class of enhancement algorithms in which a frequency weighting for an "optimum" filter is first estimated from the noisy speech, $y(n)$. The linear estimator of the undegraded speech $s(n)$, which minimizes the mean-square error is obtained by filtering $y(n)$ with a noncausal Wiener filter. This filter requires a priori knowledge of both speech and noise statistics; and must also adapt to changing characteristics. Since only a single channel exists, noise statistics must be obtained during silent frames. Also, noise-free speech is not available, therefore a priori statistics must be based upon $y(n)$, which can result in an iterative scheme. Lim and Oppenheim [38] considered such an iterative approach for additive white Gaussian noise. Their results showed improvement in speech quality for enhancement at various SNR. In addition, improvement in all-pole parameter estimation (reduced mean square error) was also observed (see fig. 16). This lead to improved intelligibility for a combined bandwidth compression system. Clements and Hansen [34,35] considered such a scheme for colored noise. Various spectral estimation techniques (MLM, MEM, Burg, Bartlett, Periodogram, PHD) were considered over a wide range of SNR (-20 to +20dB). Results using objective quality measures (Itakura-Saito, LAR, Klatt) showed improved speech quality with improved all-pole parameter estimation (see fig. 17). For the purposes of an enhancement preprocessor/recognition system, this approach would undeniably increase recognition accuracy.

Adaptive Noise Canceling and comb filtering are based on the periodicity of voiced speech. Techniques based on Wiener filtering employ a MMSE estimate criteria, and suppress noise while leaving the desired signal relatively unchanged. However, the statistics of both signals must be known a priori. LMS adaptive noise cancellation requires no a priori knowledge of the noise signal. Classical adaptive filtering (Widrow et al. [50]) assumes the input is composed of a desired speech, and unwanted, uncorrelated noise signal, along with a second reference signal consisting of noise correlated with the input noise component. Since this discussion concerns only single channel systems, most implementations of this approach may be disregarded. However, since voiced speech is highly periodic, a signal reference may be extracted for use in the LMS adaptive algorithm. The criterion for the algorithm is to form the best least-squares estimate of the clean speech signal. The success of this approach depends on the availability of a

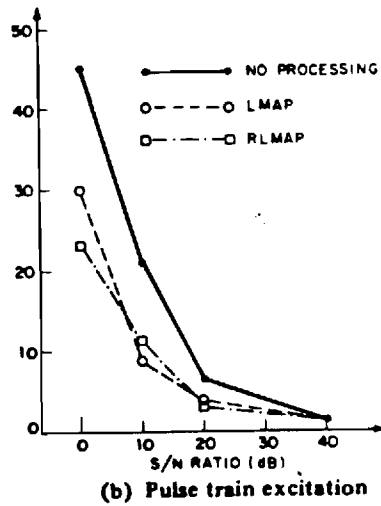
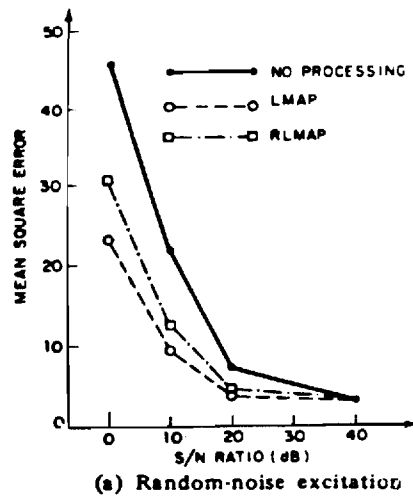


Figure 16: Performance comparison of two MAP estimation techniques for finding the all-pole LPC coefficients **a**. Both approaches use an iterative noncausal Wiener filtering scheme. Results are for estimation of all-pole parameters from noisy synthetic data. (Lim [38,40])

AVERAGE DISTORTION CLASSIFIED OVER SOUND TYPES

Performed On: NOV 29, 1985 23:01:46
By: John H. Hansen Georgia Tech.

These Measures Are Averaged Over Sentences S1, S2, S3, S4, S5, S6.

S1 P54C The pipe began to rust while new.
S2 E54C Thieves who rob friends deserve jail.
S3 T54C Add the sum to the product of these three.
S4 N54C Open the crate but don't break the glass.
S5 I54C Oak is strong and also gives shade.
S6 S54C Cats and dogs each hate the other.

Extention C = using EN.HANS_2.1C
with two spectral estimates of background noise.

Percentage
Improvement
in
Objective
Score

ITAKURA-SAITO (LIKELIHOOD) MEASURE:

Sound Type:	Number Frames	Original	Iteration					
		# 0	# 1	# 2	# 3	# 4		
Silence...	927	5.402	4.854	4.392	<u>4.347</u>	6.285	19.5	
Vowel....	698	2.506	2.238	1.952	<u>1.806</u>	2.006	27.9	
Nasal....	100	1.647	1.344	1.057	0.801	0.609	67.6	
Stop.....	293	2.402	2.112	<u>1.926</u>	2.054	3.252	19.8	
Fricative	260	1.646	1.398	<u>1.164</u>	<u>0.832</u>	1.297	49.5	
Glide.....	31	1.695	1.602	<u>1.570</u>	1.747	2.604	7.4	
Liquid....	143	4.081	3.529	2.905	2.252	<u>1.790</u>	56.1	
Affricate	20	3.037	2.569	1.979	1.285	<u>0.641</u>	78.9	
V & UV...	1545	2.422	2.126	1.851	<u>1.689</u>	2.007	30.3	
Total....	2472	3.539	3.149	2.804	<u>2.686</u>	3.611	24.1	OVERALL IMPROVEMENT

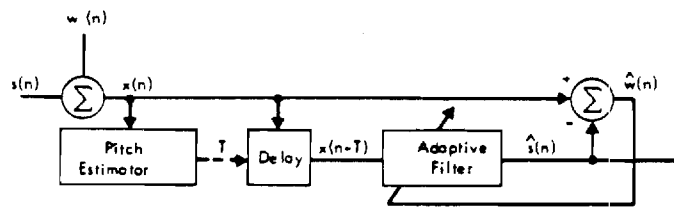
LOG AREA RATIO MEASURE:

Sound Type:	Number Frames	Original	Iteration					
		# 0	# 1	# 2	# 3	# 4		
Silence...	927	13.425	12.318	11.263	<u>10.755</u>	12.243	19.9	
Vowel....	698	8.042	7.175	6.232	5.357	<u>5.173</u>	35.7	
Nasal....	100	1.481	1.163	0.887	0.689	<u>0.658</u>	55.6	
Stop.....	293	7.960	6.800	5.643	<u>5.024</u>	6.337	36.9	
Fricative	260	9.027	7.860	6.442	4.971	<u>4.481</u>	50.4	
Glide.....	31	15.253	14.078	13.158	<u>13.051</u>	14.592	14.4	
Liquid....	143	9.671	8.834	7.966	7.174	<u>6.703</u>	30.7	
Affricate	20	5.431	4.293	3.058	1.919	<u>1.316</u>	75.8	
V & UV...	1545	8.029	7.085	6.068	<u>5.205</u>	5.266	35.2	
Total....	2472	10.053	9.047	8.016	<u>7.286</u>	7.882	27.5	OVERALL IMPROVEMENT

Figure 17: Performance evaluation of modified sequential MAP estimation technique (based on noncausal Wiener filtering), averaged over six Harvard based phonetically balanced sentences. Results indicate improvement over nine sound areas, along with total improvement. Noise degradation consisted of colored noise recorded from an aircraft cockpit. Sound types were classified by hand, and the enhancement algorithm employed two Bartlett spectral estimates during each sentence. (Hansen and Clements [34,35])

speech uncorrelated noise reference. Extracting a noise reference from the input has some disadvantages including: i) lack of stationarity for the noise, ii) not enough data to estimate the noise signal, iii) speech/silence decision is not error-free, iv) and the nonapplicability to some types of noise such as quantization noise. Sambur [66] considered an adaptive filtering technique which estimates the speech signal (using one or two pitch periods) and subtracts this from the noisy input. Variations between this approach and others rest in: i) the procedure for calculation of step size in the LMS algorithm which controls stability and rate of convergence, and ii) the filter length for the LMS algorithm. Sambur investigated the effectiveness of this approach for additive white noise and quantization noise. Results for additive noise concluded increased perceived quality for SNR of 0, 5, 10dB. Figure 18 summarizes the improvement in SNR for varying filter lengths. It was observed that the more severe the noise, the more dramatic the improvement in quality. No tests were performed to ascertain intelligibility. Quantization noise from a variable rate delta modulation system was also evaluated. The LMS adaptive filter removed some of the "granular" quality of the quantized speech. For this type of degradation, two types of noise are present, slope overload (step size too small), and granular noise (hunting due to a too large step size). Adaptive noise canceling removes the granular noise since it is signal independent and broadband but leaves slope overload noise unaffected, since it is signal dependent. Sambur also considered this scheme for an LPC analysis/synthesis system and found improved all-pole parameter estimation especially at low SNR. Boll [31] suggested a modification of this approach where the noise canceling is performed in the frequency domain. Results showed that convergence characteristics were equivalent to time domain methods and that higher quality output, free of echo resulted.

Comb filtering/Adaptive Comb filtering is similar in its basic assumptions to the LMS adaptive noise canceling technique employing a single channel. Since voiced speech is quasi-periodic, its magnitude spectrum consists of a harmonic structure. If the noise is non-periodic, its energy will be distributed throughout the spectrum. A comb filter therefore seeks to pass the voiced harmonics, and reject the noise between. A later formulation by Frazier et al. [33] showed that classical comb filtering distorts the speech somewhat. A new approach that



Adaptive filtering approach for removing noise from speech.

a)

	Input SNR		
	0 dB	5 dB	10 dB
$L = 6$	6.5	8.3	10.8
$L = 10$	6.9	8.5	10.9
$L = 14$	7.1	8.7	10.95

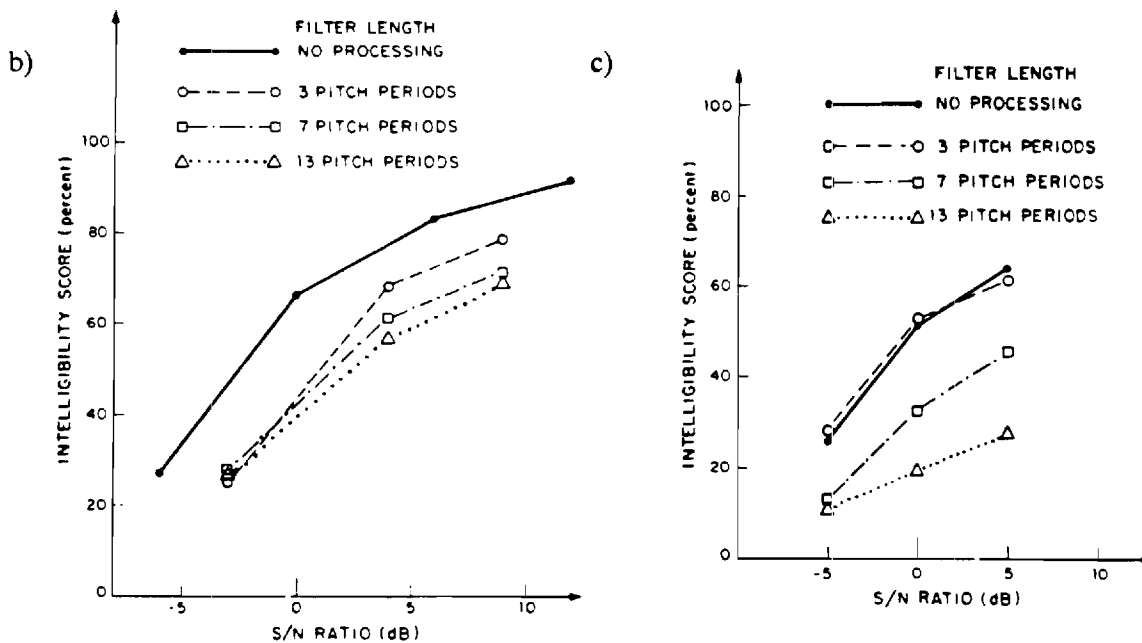


Figure 18: Adaptive Noise Canceling: a) Improvement in SNR with an additive white Gaussian noise distortion. Subjective improvement in quality was also observed, (i.e. listeners concluded that speech was more pleasant to listen to and "appeared" to have more intelligibility). However, formal quality and intelligibility tests were not undertaken. (Sambur [45])

Adaptive Comb Filtering: Intelligibility scores of Frazier's et. al., [54] filtering technique for enhancement of speech degraded by: b) competing speaker (Perlmutter et. al., [43]) and c) wide-band random noise (Lim et. al., [39]).

adapts itself both globally and locally to the time varying nature of speech was proposed. Lim et al. [39] using wide-band random noise, and Perlmutter et al. [43] using a competing speaker, evaluated this adaptive technique for varying filter length. Nonsense sentences were used in both evaluations for an intelligibility test. Figure 18 illustrates the intelligibility scores for both noise distortions. In both cases, pitch information was obtained from noise-free speech. For the competing speaker problem, results show a decrease in intelligibility. Again, with wide-band random noise, decreases in intelligibility were usually observed for various SNR's. In general, it is not realistic to assume accurate pitch information, so intelligibility scores should be lower. Even with decreases in intelligibility, both studies mention that processed speech sounded "less noisy" due to the systems' abilities to increase the SNR. No quality test were performed to verify this however.

Thus far, we have considered individual evaluations of enhancement systems. Audisio and Pirani [27] compared spectral estimation, noncausal Wiener filtering, and adaptive noise canceling in a zero mean, additive noise environment. An objective measure (the Itakura measure which is correlated with subjective quality) and intelligibility scores (Consonant Recognition Test-CRT, Preuss 1969) were obtained to evaluate the systems. Figure 19 shows Itakura distance versus SNR for voiced and unvoiced sounds. In addition, increases in percentage of intelligibility for SNR=2 dB are presented. Superior enhancement in terms of quality and intelligibility were found for spectral noise subtraction, especially in high levels of noise. Wiener filtering also resulted in improved quality and intelligibility. However, enhancement techniques (such as adaptive noise canceling or comb filtering) aimed at voiced segments had "marginal" use in increasing intelligibility. It was also observed that although unvoiced sounds (like fricatives /s/,/f/) are seriously damaged by noise disturbance, some are greatly helped by speech enhancement. Voiced sounds, such as liquids or nasals are more robust to noise, and therefore their overall contribution to intelligibility improvement has little importance. It should be noted that the particular implementations used by Audisio and Pirani may differ somewhat from others reported in the literature. Questions such as frame size, knowledge of pitch information, voiced/unvoiced decisions, choice of the algorithm analysis

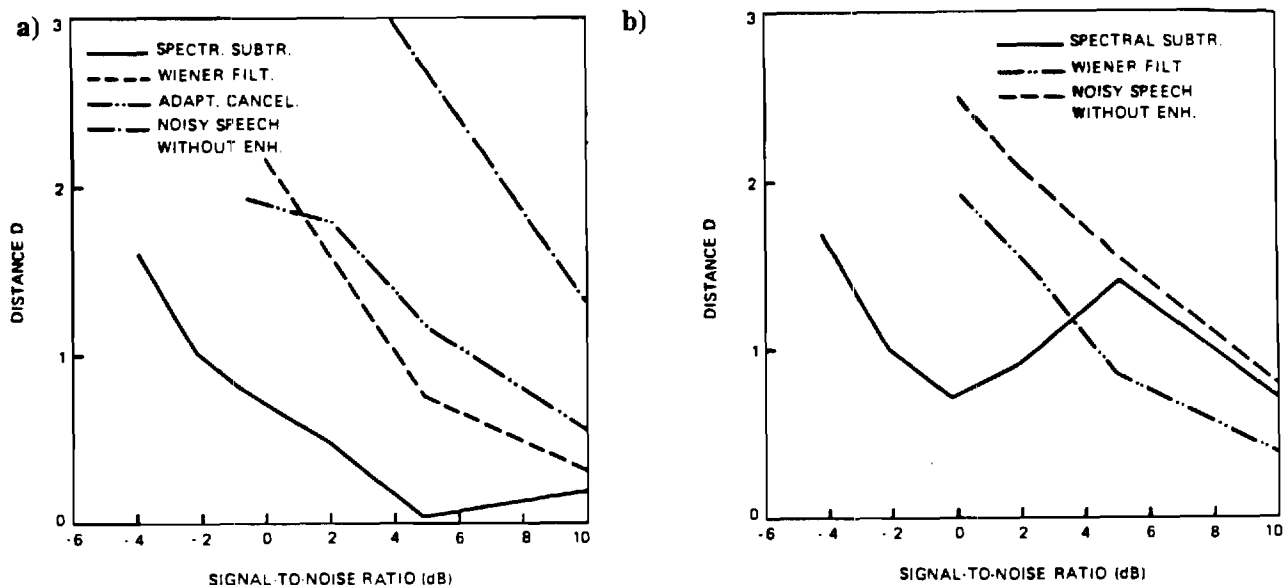


Figure 19: Comparative analysis of spectral subtraction, Wiener filtering, and adaptive filtering. a) Itakura distance versus SNR for voiced speech.

b) Itakura distance versus SNR for unvoiced sound.

c) Percentage increase in intelligibility for a SNR of 2dB.

Intelligibility scores based on the Consonant Recognition Test (CRT), which is similar to the Modified Rhyme Test (MRT). Computation of intelligibility percentage was given by this equation:

$$I = \frac{100}{T} \left\{ C - \frac{E}{N-1} \right\}$$

where: T = total number of words heard

C = number of correct answers

E = T-C = number of errors

N = number of possible choices of rhyme words offered to listener
(N=5 for this test)

parameters, and the type of noise degradation all contribute to system performance. Therefore, a comparative analysis should emphasize relative results between systems.

3.0 SUMMARY and DISCUSSION

This document has addressed several important factors for effective speech enhancement. An analysis of objectively measurable characteristics of stress and emotion in speech was considered. Thus far, research studies have shown some conflicting results. However, the general consensus is that the contour of fundamental frequency versus time to be the aspect of the speech signal which appears to provide the clearest indication of emotion or stress of a talker. Other parameters may also contribute to how a speaker conveys emotional state. These include f_0 variability, vowel/consonant durational considerations, rate of articulation, speech intensity, speech intensity variability, and possibly characteristics of the vocal tract spectrum such as formant location or overall spectral balance. There appears to be no singly reliable acoustic indicators of psychological stress or emotion. This is due in part, to the varying approaches speakers use in conveying emotional state. Other problems exist when speakers attempt to hide their emotional state. Highly visible markers/parameters are usually corrected first, leaving only subtle and unreliable markers/parameters for analysis. Prevalent trends of median f_0 and f_0 variability exist for sorrow, and fear and anger (situational stress). These variations however assume that neutral, baseline characteristics are known.

Analysis of speech enhancement algorithms was also considered. Many of these systems lead to improved quality, usually at the expense of reduced intelligibility. Those approaches that improve intelligibility, usually have devastating effects on the quality. This suggests that there remains considerable further work to be done and room for improvement. An important consideration is that evaluation of these systems is very much dependent on the context in which they are used. In some applications it is intelligibility that is of overriding importance and in others it is quality. It is intuitive, that improvement in quality is only necessary if the speech is highly intelligible to begin with. (Quality improvement of

unintelligible speech does not seem to be a fruitful research area.) Therefore, if the speech is already highly intelligible, a small decrease may be acceptable if large improvements in overall quality result. Additionally, a system may perhaps slightly reduce intelligibility but also reduce listener fatigue so that with an extended listening task intelligibility is eventually increased. Thus far, it appears that none of the systems discussed have been examined for their potential in reducing listener fatigue. Reducing listener fatigue in a two way communication system (aircraft cockpit environment) may also contribute to a reduced level of situational stress.

4.0 REFERENCES

4.1 STRESS and EMOTION in SPEECH

- [1] A.M. Aull, V.W. Zue, " Lexical Stress Determination and its Application to Large Vocabulary Speech Recognition, " *Proc. 1985 IEEE ICASSP*, pp. 41.1.1-4, Mar.1985.
- [2] A.J. Bachrach, " Speech and its Potential for Stress Monitoring: Monitoring Vital Signs in the Diver, " Naval Medical Research Institute TECHNICAL REPORT, pp.78-93, August 1979.
- [3] J.K. Darby, *Speech Evaluation in Psychiatry*, Grune & Stratton, New York, New York, 1981.
- [4] L. Goldberger, S. Breznitz, *Handbook of Stress: Theoretical & Clinical Aspects*, Free Press, Macmillan Publishing, New York, New York, 1982.
- [5] M.H.L. Hecker, K.N. Stevens, etal, " Manifestations of Task-Induced Stress in the Acoustic Speech Signal, " *Journal of the Acoustical Society of America*, Vol. 44, No.4, pp. 993-1001, April 1968.
- [6] J.W. Hicks, H. Hollien, " The Reflection of Stress in Voice-1: Understanding the Basic Correlates, " *The 1981 CARNAHAN Conf. on Crime Countermeasures*, pp. 189-195, May 1981.
- [7] H. Hollien, J.W. Hicks, " The Reflection of Stress in Voice-2: The Special Case of Psychological Stress Evaluators, " *The 1981 CARNAHAN Conf. on Crime Countermeasures*, pp. 196, May 1981.
- [8] I. Kuroda, etal, " Method for Determining Pilot Stress Through Analysis of Voice Communication, " *Aviation,Space, and Environmental Medicine*, pp. 528-533, May 1976.
- [9] P. Lieberman, *Intonation, Perception, and Language*, MIT Press, Cambridge, Massachusetts, 1967.
- [10] P. Lieberman, S. Michaels, " Some Aspects of Fundamental Frequency and Envelope Amplitude as Related to the Emotional Content of Speech, " *Journal of the Acoustical Society of America*, Vol. 34, No.7, pp. 922-927, July 1962.
- [11] F.J. Malkin, K.A. Christ, " Human Factors Engineering Assessment of Voice Technology for the Light Helicopter Family, " U.S. Army Human Engineering Laboratory Technical Report, pp. 1-20, June 1985.
- [12] L.H. Nakatani, C.H. Aston, " Acoustic and Linguistic Factors in Stress Perception, " *BELL System Technical Report*, October 1978.
- [13] J.B. Peckham, " A Device For Tracking The Fundamental Frequency of Speech and its Application in the Assessment of 'Strain' in Pilots and Air Traffic Controllers, " Technical Report 79056, Royal Aircraft Establishment, pp.1-55, May 1979.

- [14] J.M. Pickett, *The Sound of Speech Communication*, University Park Press, Baltimore, Maryland, 1980.
- [15] D.B. Pisoni, et al, " Some Acoustic-Phonetic Correlates of Speech Produced in Noise, " *Proc. 1985 IEEE ICASSP*, pp. 41.10.1-4, March 1985.
- [16] G.K. Poock, J.W. Armstrong, " Effect of Operator Mental Loading on Voice Recognition Systems Performance, " Naval Postgraduate School Technical Report, pp. 1-60, August 1981.
- [17] G.K. Poock, J.W. Armstrong, " Effect of Task Duration on Voice Recognition System Performance, " Naval Postgraduate School Technical Report, pp. 1-71, September 1981.
- [18] L. Reed, " Military Applications of Voice Technology, " *Speech Technology*, pp. 42-50, February 1985.
- [19] D. Rostolland, " Acoustic Features of Shouted Voice Part I, " *Acustica*, Vol. 50, pp. 118-125, 1982.
- [20] D. Rostolland, " Phonetic Structure of Shouted Voice Part II, " *Acustica*, Vol. 51, pp. 80-89, 1982.
- [21] M.J. Russell, R.K. Moore, " Recordings Made For Automatic Speech Recognition Assessment and Research, " Royal Signals and Radar Establishment Technical Report, 1983, AD-A146 824.
- [22] K.R. Scherer, D.R. Ladd, " Vocal cues to speaker affect: Testing two models, " *Journal of the Acoustical Society of America*, Vol. 76, No.5, pp. 1346-1356, November 1984.
- [23] P.V. Simonov, M.V. Frolov, " Analysis of the Human Voice as a Method of Controlling Emotional State: Achievements and Goals, " *Aviation, Space, and Environmental Sciences*, pp. 23-25, January 1977.
- [24] C.A. Simpson, " Speech Variability Effects on Recognition Accuracy Associated With Concurrent Task Performance by Pilots, " Psycho-Linguistic Research Associates Technical Report, pp. 1-15, April 1985.
- [25] L.A. Streeter, N.H. Macdonald, et al, " Acoustic and Perceptual Indicators of Emotional Stress, " *Journal of the Acoustical Society of America*, Vol. 73, No.4, pp. 1354-1360, April 1983.
- [26] C.E. Williams, K.N. Stevens, " Emotions and Speech: Some Acoustical Correlates, " *Journal of the Acoustical Society of America*, Vol. 52, No.4, pp. 1238-1250, 1972.

4.2 SPEECH ENHANCEMENT SYSTEMS: A Comparative Analysis

- [27] G. Audisio, G. Pirani, " Noisy Speech Enhancement: A Comparative Analysis, " CSELT Technical Reports, Vol. XII, No. 6, pp. 607-615, Dec. 1984.
- [28] M. Berouti, J. Makhoul, R. Schwartz, " Enhancement of Speech Corrupted by Acoustic Noise, " *ICASSP 79*, pp. 208-211, 1979.

- [29] S.F. Boll, " Suppression of Noise in Speech Using the SABER Method, " *ICASSP 78*, pp. 606-609, 1978.
- [30] S.F. Boll, " Suppression of Acoustic Noise in Speech Using Spectral Subtraction, " *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-27, pp. 113-120, April 1979.
- [31] S.F. Boll, " Adaptive Noise Canceling in Speech Using the Short-Time Transform, " *ICASSP 80*, pp. 692-695, 1980.
- [32] E.R. Ferrara, B. Widrow, " Multichannel Adaptive Filtering for Signal Enhancement, " *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-29, No.3, pp. 766-775, June 1981.
- [33] R.H. Frazier, A.V. Oppenheim, et al, " Enhancement of Speech by Adaptive Filtering, " *ICASSP 76*, pp. 251-253, 1976.
- [34] J.H. Hansen, M.A. Clements, " Enhancement of Speech Degraded by Non-White Additive Noise, " Final Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [35] J.H. Hansen, M.A. Clements, " Objective Quality Measures Applied to Enhanced Speech," *Conf. of the Acoustical Society of America*, 110th Meeting, C11, Nashville, Tenn., Nov. 1985.
- [36] B.A. Hanson, D.Y. Wong, " The Harmonic Magnitude Suppression (HMS) Technique for Intelligibility Enhancement in the Presence of Interfering Speech, " *ICASSP 84*, 18A.5.1-4, March 1984.
- [37] J.S. Lim, " Evaluation of a Correlation Subtraction Method For Enhancing Speech Degraded by Additive White Noise," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, pp. 471-472, Oct. 1978.
- [38] J.S. Lim, A.V. Oppenheim, " All-Pole Modeling of Degraded Speech," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, pp. 197-210, June 1978.
- [39] J.S. Lim, A.V. Oppenheim, L.D. Braida, " Evaluation of an Adaptive Comb Filtering Method for Evaluating Speech Degraded by White Noise Addition," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, No.5, pp. 354-358, Aug. 1978.
- [40] J.S. Lim, A.V. Oppenheim, " Enhancement and Bandwidth Compression of Noisy Speech," *Proceedings of the IEEE*, Vol. 67, pp. 1586-1604, Dec. 1979.
- [41] D. Malah, R.V. Cox, " A Generalized Comb Filtering Technique for Speech Enhancement," *ICASSP 82*, pp. 160-163, 1982.
- [42] R.J. McAulay, M.L. Malpass, " Speech Enhancement Using a Soft-Decision Noise Suppression Filter," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, No.2, pp. 137-145, April 1980.
- [43] Y.M. Perlmuter, L.D. Braida, et al, " Evaluation of a Speech Enhancement System, " *ICASSP 77*, pp. 212-215, 1977.
- [44] T.L. Peterson, S.T. Boll, " Acoustic Noise Suppression in the Context of a Perceptual Model, " *ICASSP 81*, pp. 1086-1088, 1981.

- [45] M.R. Sambur, "LMS Adaptive Filtering for Enhancing the Quality of Noisy Speech," *ICASSP 78*, pp. 610-613, 1978.
- [46] C.K. Un, K.Y. Choi, "Improving LPC Analysis of Noisy Speech by Autocorrelation Subtraction Method, " *ICASSP 81*, pp. 1082-1085, 1981.
- [47] D.L. Wang, J.S. Lim, " The Unimportance of Phase in Speech Enhancement, " *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-30, No.4, pp. 679-681, Aug. 1982.
- [48] M.R. Weiss, E. Aschkenasy, " Computerized Audio Processor, " Final Report Rome Air Development Center RADC-TR-83-109, May 1983.
- [49] M.R. Weiss, E. Aschkenasy, T.W. Parsons, " Study and development of the INTEL technique for improving speech intelligibility, " Nicolet Scientific Corp., Final Report NSC-FR/4023, Dec. 1974.
- [50] B. Widrow, J.M. McCool, etal, " Adaptive Noise Canceling: Principles and Applications, " *Proceedings of the IEEE*, Vol. 63, No.12, pp. 1692-1716, Dec. 1975.

APPENDIX: Speech Enhancement Systems

The problem of enhancing speech degraded by additive background noise has received much attention since the mid to late 1970's. Many approaches have been taken, each attempting to capitalize on various characteristics or constraints, all with varying degrees of success. One application for such systems is as a preprocessor for speech recognition or bandwidth compression. Assumptions concerning noise characteristics and assumed speech model effect the formulation of these algorithms. Three assumptions made for most systems are: i) distorting noise is additive, ii) noise and speech are uncorrelated, iii) the only input available is the distorted speech signal. Speech enhancement systems are classified into two broad classes: those based on the speech signal being a stochastic process, and those based on perceptual aspects of speech. Systems based on the signal being a stochastic process rely on a given mathematical criterion. Systems based on perceptual criteria attempt to improve aspects important in human perception. For example, one technique may concentrate on improving the quality of consonants, since consonants are known to be important for intelligibility though they represent only a small percentage of overall signal energy. This appendix will briefly review three classes of speech enhancement. The first class concentrates its analysis in the short-time spectral domain. These techniques suppress noise by subtracting an estimated noise bias (spectral or autocorrelation) found during non-speech activity. Such systems include spectral subtraction, correlation subtraction, and the INTEL system (cepstral domain). The second class of systems is based on an all-pole modeling technique which incorporates noncausal Wiener filtering. This approach requires a priori knowledge of the noise and speech statistics and will eventually result in an iterative enhancement scheme. The last class of systems are based on the periodicity of voiced speech. These systems include adaptive noise canceling based on the LMS algorithm, and comb filtering/adaptive comb filtering.

SPECTRAL SUBTRACTION and CORRELATION SUBTRACTION

The process of spectral subtraction involves suppressing noise from speech by subtracting a spectral noise bias found during non-speech activity. Specifically, this technique capitalizes on the short-time spectrum of speech. Since the human auditory system is relatively insensitive to phase distortion, this approach deals only with the magnitude of the short-time spectrum. Suppose for example, a stationary signal $x(n)$ has been degraded by an additive, uncorrelated noise source $d(n)$, with a known power spectrum. The power spectrum of the original signal can be estimated by the spectral subtraction process. The additive distorted time waveform can be represented as:

$$y(n) = x(n) + g d(n) \quad (A.1)$$

also, let the power spectra of each component be represented as $P_Y(j\omega)$, $P_X(j\omega)$, and $P_D(j\omega)$. This results in equation (A.2).

$$P_Y(j\omega) = P_X(j\omega) + g P_D(j\omega) \quad (A.2)$$

Therefore, to estimate $P_X(j\omega)$, the algorithm simply subtracts from $P_Y(j\omega)$, an estimate of the noise spectrum $P_D(j\omega)$. For the general speech waveform case, the signal $x(n)$ is not truly stationary. However, it is safe to assume a short-time stationary property, due to the limitations of the speech production system. This requires the spectral subtraction process to be represented using short-time power spectra. The windowing process results in the following relation in the magnitude spectral domain:

$$\begin{aligned} |Y_w(j\omega)|^2 &= |S_w(j\omega)|^2 + |D_w(j\omega)|^2 \\ &\quad + S_w(j\omega) D_w^*(j\omega) + S_w^*(j\omega) D_w(j\omega) \end{aligned} \quad (A.3)$$

The term, $|S_w(j\omega)|^2$ is referred to as the short-time magnitude spectrum of speech. To carry out the spectral subtraction technique, equation (A.3) must be solved. Since the windowed noise

data $y_w(n)$ is directly available. $|Y_w(j\omega)|^2$ can be formed directly. The terms on the right hand side, $|D_w(j\omega)|^2$, $S_w(j\omega) D_w^*(j\omega)$, and $S_w^*(j\omega) D_w(j\omega)$ cannot be expressed exactly. Therefore, spectral subtraction must employ their estimates, namely $E[|D_w(j\omega)|^2]$, $E[S_w(j\omega) D_w^*(j\omega)]$, and $E[S_w^*(j\omega) D_w(j\omega)]$. Note that $E[\]$ denotes the expectation operation. Since $d(n)$ is assumed zero mean, and $s(n)$ and $d(n)$ uncorrelated, the latter two expectations are zero. Therefore, the resulting spectral subtraction relation is,

$$|\hat{S}_w(j\omega)|^2 = |Y_w(j\omega)|^2 - E[|D_w(j\omega)|^2] \quad (A.4)$$

At this point, it is obvious why such an approach would be detrimental for an application such as echo canceling, due to the similar characteristics between speech and subsequent echoes (i.e. expectation between cross terms cannot be ignored). The expected noise power spectrum in (A.4) can be found either from measurements during non-speech activity, or a second input consisting of noise only data representing a sample function of the noise process. One point which can be observed from equation (A.4), is that $|\hat{S}_w(j\omega)|^2$ is not guaranteed to be non-negative. This is due to the possibility of the noise spectrum exceeding the noisy speech spectrum. Since a magnitude spectrum is always positive, negative results must be altered. This can be accomplished using half-wave rectification, full-wave rectification, or a weighted subtraction, etc. Most techniques use half-wave rectification (set negative portions to zero).

Once the spectral magnitude $|\hat{S}_w(j\omega)|$ is found, a variety of techniques are available for reconstructing the time waveform $s_w(n)$. A system proposed by Boll [29,30] attempts to reduce spectral error by applying three processing steps once the spectral magnitude has been found. The spectral error can be defined as follows,

$$\epsilon(j\omega) = \hat{S}_w(j\omega) - S(j\omega) \quad (A.5)$$

The three steps which Boll formulates are magnitude averaging, half-wave rectification, and residual noise reduction. The process of magnitude averaging reduces spectral error by

performing local averaging of the spectral magnitudes, (i.e. instead of $|X_w(j\omega)|$, $\overline{|X_w(j\omega)|}$ is used). The magnitude averaged spectrum is found using the sample mean in equation (A.6).

$$\overline{|X_w(j\omega)|} = \frac{1}{M} \sum_{i=0}^{M-1} |X_i(j\omega)| \quad (A.6)$$

Here, $|X_i(j\omega)|$ represents the i^{th} time-window power spectrum of the waveform $x(n)$. Therefore the resultant estimator, using the noisy phase $\Theta_Y(j\omega)$ from the original distorted speech with magnitude averaging, is of the form

$$\hat{S}_{MA}(j\omega) = \{ \overline{|Y_w(j\omega)|} - E[|D_w(j\omega)|] \} e^{j\Theta_Y(j\omega)} \quad (A.7)$$

The magnitude averaging method works well if the time waveform is stationary. Unfortunately, the value of M in equation (A.6) is limited by the short-time stationarity assumption. Therefore, only a few frames of data can be used in averaging. Boll's second processing step is half-wave rectification which reduces the mean noise level by an amount $E[D_w(j\omega)]$. With this, low variance coherent noise is approximately eliminated. The disadvantage with half-wave rectification is that it is possible for the speech plus noise spectrum to be less than $E[D_w(j\omega)]$ and consequently, speech information is removed. The last step in Boll's algorithm is residual noise reduction. After half-wave rectification, the spectral bands of speech plus noise above the threshold $E[D_w(j\omega)]$ remain, thereby preserving a residual noise component. The argument at this point is that residual noise can be reduced by replacing the present frame value with a minimum value from adjacent frames. The question which arises is why such a method should work? The answer is as follows. If $|\hat{S}_w(j\omega)|$ is less than the maximum noise residual, and if it varies from frame-to-frame; then there is a high probability that the spectrum at that frequency is due to noise. Therefore, the noise can be suppressed by taking the minimum from adjacent frames. If $|\hat{S}_w(j\omega)|$ is less than the maximum noise residual, but $|\hat{S}_w(j\omega)|$ is approximately constant between adjacent frames; then a high probability exists for the spectrum at that frequency to be low energy speech. Therefore, taking the minimum will not affect the information content. Finally, if $|\hat{S}_w(j\omega)|$ is greater than the maximum noise residual, then speech is present in the signal at that frequency; therefore subtracting off the

noise bias is enough. Boll evaluated this algorithm for helicopter speech. Results showed that spectral subtraction alone does not decrease intelligibility, but does increase quality, especially in the areas of increased pleasantness and inconspicuousness of noise background.

An interesting point to be made is that spectral subtraction can also be interpreted as estimating the short-time correlation function $\phi_s(n)$ as follows,

$$\phi_s(n) = \phi_y(n) - E[\phi_d(n)] \quad (\text{A.8})$$

Where the windowed autocorrelation function for the estimated speech is defined as,

$$\begin{aligned} \phi_s(n) &= \sum_{k=-\infty}^{\infty} s_w(k) s_w(k-n) \\ &= \mathcal{F}^{-1}[|S_w(j\omega)|^2] \end{aligned} \quad (\text{A.9})$$

with $\mathcal{F}^{-1}[\]$ defined as the inverse Fourier transform operation. The relations for the original distorted speech correlation function $\phi_y(n)$, and the estimated noise correlation function $E[\phi_d(n)]$ are defined similarly. For this reason, the spectral subtraction technique is sometimes referred to as the correlation subtraction technique. Both methods simply perform their noise reduction procedures in different domains. Figure 15 illustrates various implementations of spectral subtraction and correlation subtraction.

INTEL SYSTEM

Another area of speech enhancement which has received much attention is in the development of noise suppression prefilters. In this approach a spectral decomposition of a frame of noisy speech is performed and a particular spectral line is attenuated depending on how much the measured speech plus noise power exceeds an estimate of the background noise power. One approach in particular developed by Weiss and Aschkenasy [48,49], implements a real-time audio processor using the INTEL system and a tone component suppression filter.

Weiss and Aschkenasy originally developed the INTEL [48] as a generalization of spectral subtraction. (Figure 15 illustrates the differences between the two.) This involved raising the noisy speech magnitude spectrum to a power α . Later work [49], resulted a real-time filter that removes interference from received or recorded data, termed *Computerized Audio Processor*. The system is made up of two processing sections; the first called *Digital Spectral Shaping* (DSS) detects and attenuates impulsive and tonal noise; the second is INTEL, which is used to attenuate additive wideband random noise.

Digital Spectral Shaping is a relatively simple processing step which removes long tonal noises from the input signal. To be effective, three steps must be accomplished. First, the tone must be detected accurately. Next, the system must remove the maximum amount of tone energy once it has been detected while removing a minimum amount of speech energy. The last step requires that the regenerated speech be maximally free of discontinuities and distortion. The detection process rests on exploiting differences between speech and noise. Tone noise has a greater stability in both frequency and amplitude as opposed to the assumed quasi-stationary model for speech. Consider the amplitude spectrum of the tone noise. Spectral peaks at the frequency of the tone should be observed. The speech spectrum instead, will be smooth over the entire frequency band with smooth peaks at the formant frequencies and finite nonzero bandwidths. To minimize the speech versus noise overlap in the frequency domain, tone energy should be concentrated into as narrow a spectrum as possible. Figure 1A illustrates the process for this processing section.

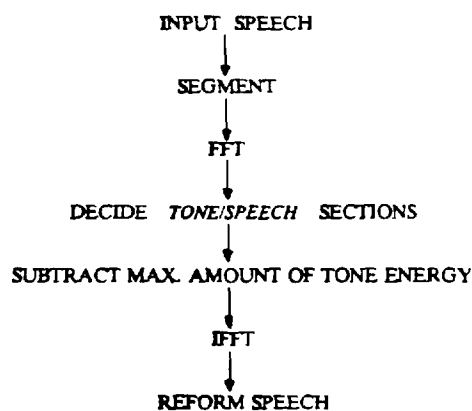


Figure 1A: Tone removal for Computerized Audio Processor.

To accomplish a minimum between speech spectrum and tone peak overlap, an appropriate weighting function on the time series must be chosen. Choice of analysis frame length must also be considered in the segmentation portion. This particular system uses a frame length of 200 msec, with a Bartlett window overlapped by 50%. This approach for tone removal works well when tone frequencies are different from the speech spectrum. The more profound the difference, the more successful the procedure becomes. This approach is relatively useless if the tone component is random, thus requiring the second processing section.

The primary use of INTEL is to attenuate additive wideband random noise. The input signal is transformed to a modified cepstrum domain. An estimate of the noise cepstrum is subtracted, and the resulting cepstral data is used to reform the enhanced speech. (The term modified cepstrum is used since the true cepstrum is the power spectrum of the log amplitude spectrum.) Figure 2A illustrates the INTEL procedure. (Note that the power term a has been set to $1/2$ in this implementation.)

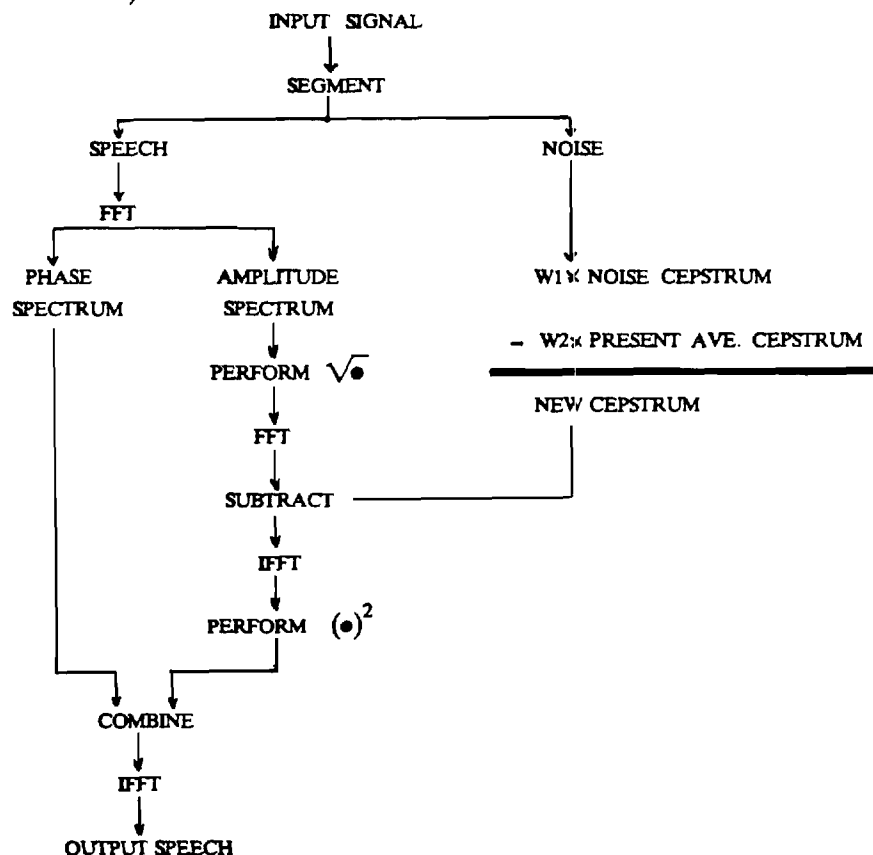


Figure 2A: INTEL Procedure for removing additive wideband random noise.

Figure 2A shows that the difference between INTEL and spectral subtraction is the added transform pair, thereby performing the subtraction operation in the modified cepstrum domain. Improvement over tone subtraction results due to differences in cepstrum characteristics between the speech and random noise. Above a quefrency of 0.5 msec, the noise energy falls off quickly while speech energy is still present at pitch period and its harmonics. Therefore, if a noise only cepstrum could be found, subtracting it from the speech plus noise cepstrum should greatly reduce the broadband random noise. How to compute the noise only cepstrum must therefore be addressed. To accomplish this, a 'lossy moving average' of the noise cepstrum is formed. The noise cepstrum is then able to follow changes in the noise distribution. The two weights for the present and update noise cepstra must be chosen during processing. Once the algorithm is able to track the noise cepstrum, the choice of scale factor for subtraction between noisy speech and noise only cepstra must be chosen. Weiss and Aschkenasy concluded that three scale factors were adequate for processing. The process being carried out as such:

$$\begin{array}{c} \text{SPEECH PLUS NOISE CEPSTRUM} \\ - s_i \text{ [AVE. NOISE CEPSTRUM]} \\ \hline \text{ESTIMATED SPEECH ONLY CEPSTRUM} \end{array}$$

With the three scale factors s_i defined as follows;

s_0 for 0.0 to 0.1 msec

s_1 for 0.1 to 0.5 msec

s_2 for 0.5 to ∞ msec

Results indicate that choice of $\{s_i\}$ is dependent on SNR, but somewhat independent of particular noise distribution. Their system used two sets of scale factors; one for use above 6dB, the second for below 0dB. Between 0 and 6dB, either set produced similar results.

Though this system is useful for removing wideband random noise, some disadvantages do exist. Since the average noise cepstrum is built up over time, any long sections of silence (1

sec or more) would drive the average noise cepstrum to zero. When the signal should reappear, a loud noise burst results until the noise cepstrum can be built up. Choice of update weights for the noise cepstrum must also be selected. This choice depends on the speed with which the system tracks changes in the noise characteristics. Scale factors for subtraction of the cepstra must also be chosen in some optimal fashion. Although Weiss and Aschkenasy found these to be somewhat independent a particular noise distribution, this may not be true for all types of noise. Although no quantitative results were presented in this investigation, it was suggested that some improvement in intelligibility is possible. This is dependent on how accurately the noise cepstrum is able updated from silent frames.

WIENER FILTERING/ALL-POLE MODELING

The estimation of speech parameters in an all-pole model for the additive white gaussian noise case was investigated by Lim and Oppenheim [38], and later for colored noise degradation by Hansen and Clements [34,35]. From these studies, it was shown that the estimation procedures which result in linear equations without background noise, become nonlinear when noise is introduced. However, by allowing a suboptimal procedure to be taken, an iterative algorithm results which possesses the property that the estimation procedure is linear at each iteration.

Consider the statistical parameter estimation of speech in the absence of noise. In general, a speech waveform can be modeled by a linear short-time stationary system. The transfer function $V(z)$ of this all-pole linear system is of the form:

$$V(z) = \frac{1}{1 - \sum_{k=1}^P a_k z^{-k}} \quad (\text{A.10})$$

Over a short-time basis, the speech can be represented as the following difference equation

$$s(n) = \sum_{k=1}^P a_k s(n-k) + u(n) + e(n) \quad (\text{A.11})$$

where the a_k 's are the predictor coefficients, $u(n)$ is the input excitation and $e(n)$ represents the residual error between the speech waveform and response from the all-pole model driven by

the input excitation. An assumption which usually made is to lump the input excitation and error residual into a single noise excitation represented as $g w(n)$. Here g represents a gain factor, $w(n)$ represents a white gaussian noise signal with zero mean and unit variance. Assuming a random process as the excitation, the speech signal may be represented as

$$s(n) = \mathbf{a}^T s(n-1, n-p) + g w(n) \quad (\text{A.12})$$

This indicates that there are $2p + 1$ unknowns including; i) all-pole predictor coefficients $\mathbf{a}^T = [a_1, a_2, \dots, a_p]$, ii) initial conditions for the p^{th} order predictor given by $S_I = s(-1, -p)$, iii) and the gain factor g for the input excitation, which must be determined to recover $s(n)$. Consider the case where all unknown parameters are random with a priori Gaussian probability density functions. The procedure used is a maximum a priori (MAP) estimator, which maximizes the probability density function of the parameters given the observations. Using the model of equation A.12, and the observation vector $\mathbf{S}_O = s(1, N)$, the predictor coefficients \mathbf{a} are to be estimated. Therefore, MAP estimation corresponds to maximizing $p(\mathbf{a}|\mathbf{S}_O)$, which in general requires the solution of a set of nonlinear equations for the AWGN (additive white Gaussian Noise) case.

Four basic approaches may be taken in attempting to estimate \mathbf{a} , without solving a set of nonlinear equations. These approaches are discussed in greater detail in the references [55,59]. The first method involves the joint estimation all the parameters, assuming no a priori information on their statistics. This is equivalent to maximizing the probability density function $p(\mathbf{a}, g, S_I | \mathbf{S}_O)$ with respect to \mathbf{a}, g, S_I assuming no prior knowledge of \mathbf{a}, g, S_I . This method leads to the same linear equations obtained by the covariance method of LPC. Unfortunately, this estimation approach degrades quickly in the presence of additive background noise.

Another method, is to assume that S_I is known, and jointly estimate \mathbf{a} and g assuming no a priori information. This is the same as maximizing the probability density function $p(\mathbf{a}, g | \mathbf{S}_O, S_I)$ with respect to \mathbf{a} and g which also leads to a set of linear equations. Depending

on how the initial condition vector S_I is chosen, the approach reduces to either the correlation or covariance methods of LPC (equations A.13, A.14).

$$\begin{aligned} \text{Covariance Method} \quad & \sum_{n=P}^{N-1} [s(n) - \mathbf{a}^T s(n-1, n-p)] s(n-i) = 0 \\ & \text{for } i = 1, 2, \dots, P \end{aligned} \quad (\text{A.13})$$

$$\text{Correlation Method} \quad \sum_{n=0}^{N+P-1} [s(n) - \mathbf{a}^T s(n-1, n-p)] s(n-i) = 0 \quad (\text{A.14})$$

The third approach is to assume the gain term g is known and jointly estimate \mathbf{a} and S_I assuming no a priori information. Again, this can be viewed as maximizing the probability density function $p(\mathbf{a}, S_I | S_O; g)$ with respect to \mathbf{a} and S_I .

The last method is to estimate \mathbf{a} only. This appears to be the logical choice since it is the predictor coefficients \mathbf{a} which are desired. This estimation procedure is performed by maximizing the probability density function $p(\mathbf{a} | S_O)$ with respect to \mathbf{a} , which results in a set of nonlinear equations in \mathbf{a} . However, in the special case where S_I and g are known, the equations become linear; hence the reason for including the first three estimation procedures. Therefore, this special case suggests an overall procedure which combines two of these methods in order to reduce the nonlinear system of equations into a linear set. Consider a procedure, which incorporates the first method for estimating g and S_I ; followed by the fourth technique assuming the estimates of g and S_I are exact, and maximize $p(\mathbf{a} | S_O; g, S_I)$ with respect to \mathbf{a} . This procedure works well in the noise-free case, but degrades in the presence of noise. Nevertheless, this procedure does suggest a method for obtaining the LPC predictor coefficients \mathbf{a} , even if they are corrupted versions of the noise-free case. As a side note, if $p(\mathbf{a} | S_O; g, S_I)$ is assumed gaussian, and symmetric about the conditional mean $E[\mathbf{a} | S_O; g, S_I]$, then MAP estimation of \mathbf{a} is equivalent to a minimum mean squared error estimate (MMSE) of \mathbf{a} .

Thus far, MAP estimation of the LPC coefficients \mathbf{a} have been considered in the absence of noise. These methods which require a solution of a set of linear equations, become nonlinear when noise is introduced. Therefore, a "suboptimal" approach was developed which results in linear at each iteration. Using the same speech model, a background noise term can now be

included. With this modification, the observation vector becomes;

$$\mathbf{Y}_O = y(N-1,0) = s(N-1,0) + d(N-1,0) \quad (\text{A.15})$$

where $s(N-1,0)$ are N samples of the original speech, and $d(N-1,0)$ represents the additive background noise. Two assumptions on \mathbf{d} are considered; first, that $d(n)$ and $s(n)$ are uncorrelated; the second, that $d(n)$ is white gaussian with zero mean and variance σ_d^2 . By following a similar procedure from the noise-free case, choose \mathbf{a} to maximize $p(\mathbf{a}|\mathbf{Y}_O)$. Note that the estimate is now conditioned on noisy observations \mathbf{Y}_O , not the original \mathbf{S}_O . Substituting the degraded speech into the assumed speech model gives the following equation for the observation vector;

$$\mathbf{Y}_O = \mathbf{a}^T y(n-1,n-p) + g w(n) + d(n) - \mathbf{a}^T d(n-1,n-p) \quad (\text{A.16})$$

Using this expression, represent $p(\mathbf{Y}_O|\mathbf{a},g,\mathbf{S}_I)$ as a product of three terms, namely;

$$\begin{aligned} p(\mathbf{Y}_O|\mathbf{a},g,\mathbf{S}_I) &= \prod_{n=p}^{N-1} p\{y(n)|\mathbf{a},g,\mathbf{S}_I,y(n-1,0)\} \\ &\quad \times \prod_{n=1}^{p-1} p\{y(n)|\mathbf{a},g,\mathbf{S}_I,y(n-1,0)\} \\ &\quad \times p\{y(n)|\mathbf{a},g,\mathbf{S}_I\} \end{aligned} \quad (\text{A.17})$$

If $n > p$, as in the first product term, the probability density function $p(y(n)|\mathbf{a},g,\mathbf{S}_I,y(n-1,0))$ will be gaussian with mean and variance given by

$$\begin{aligned} \text{mean} &= \mathbf{a}^T y(n-1,n-p) - \mathbf{a}^T E[*] \\ \text{variance} &= g^2 + \sigma_d^2 + \mathbf{a}^T \text{VAR}[*] \mathbf{a} \end{aligned} \quad (\text{A.18})$$

with,

$$\begin{aligned} E[*] &= E[d(n-1,n-p)|\mathbf{a},g,\mathbf{S}_I,y(n-1,0)] \\ \text{VAR}[*] &= \text{VAR}[d(n-1,n-p)|\mathbf{a},g,\mathbf{S}_I,y(n-1,0)] \end{aligned}$$

As can be seen, $E[*]$ and $\text{VAR}[*]$ are the mean and variance of $d(n-1,n-p)$ conditioned on $\mathbf{a},g,\mathbf{S}_I,y(n-1,0)$. Since $E[*]$ and $\text{VAR}[*]$ are both functions of the predictor coefficients \mathbf{a} , the resulting equations for maximizing $p(\mathbf{a}|\mathbf{Y}_O)$ are nonlinear; involving partial derivatives with

respect to \mathbf{a} . Similar results are found for the other three approaches, since the conditional probability density functions consist of products of several terms which include;

$$\prod_{n=P}^{N-1} p(y(n)|\mathbf{a}, g, S_I, y(n-1, 0)),$$

where the dependence on \mathbf{a} is evident. To alleviate this obstacle, Lim and Oppenheim considered a suboptimal solution based on the noise free case. It was previously observed that the estimation of \mathbf{a} from S_O assuming S_I and g resulted in a linear set of equations. This suggests a two step approach based on MAP estimation of S_O given Y_O , followed by MAP estimation of \mathbf{a} given \hat{S}_O , where \hat{S}_O is the result of the first estimation. With this approach, the algorithm can be summarized as shown in figure 3A.

- i. Begin with an assumed set of predictor coefficients, \mathbf{a}_0 .
- ii. Estimate S_O by maximizing $p(S_O|\mathbf{a}, Y_O; g, S_I)$. The output is: \hat{s}_{O_i}
- iii. Given \hat{s}_{O_i} , re-estimate \mathbf{a} . The output is: $\hat{\mathbf{a}}_i$.
- iv. Continue until $\hat{\mathbf{a}}_i$ approaches $\hat{\mathbf{a}}_\infty$.

Figure 3A: A Two Step MAP Estimation Algorithm for Speech Enhancement

Observations indicate that this algorithm converges to a local maximum of the joint density $p(\mathbf{a}, S_O|Y_O; g, S_I)$. In particular, if the probability density function is unimodal, and the initial estimate for \mathbf{a} is such that the local maximum equals the global maximum, then the procedure is equivalent to the joint MAP estimate of \mathbf{a} and S_O . The implementation of this approach still requires some simplification. The two MAP estimates which must be performed are;

$$\text{i. } \text{MAX}_{\mathbf{a}} p(\hat{\mathbf{a}}_i | \hat{S}_{O_i}, Y_O; g, S_I) \quad \text{Which gives } \hat{\mathbf{a}}_i \quad (\text{A.19})$$

$$\text{ii. } \text{MAX}_{S_O} p(S_O | \hat{\mathbf{a}}_i, Y_O; g, S_I) \quad \text{Which gives } \hat{s}_{O_i} \quad (\text{A.20})$$

The first estimate requires the solution of P linear equations to obtain $\mathbf{a}^T = [a_1, \dots, a_P]$. The second requires the solution of N linear equations for $S_O^T = [s_0, s_1, s_2, \dots, s_{N-1}]$. Since N may be

of the order of several hundred samples, the second estimation becomes computationally intensive. To alleviate this, the equation can be rewritten in a simplified form. Using Baye's rule, the gaussian probability density function for $p(Y_O|a_i, S_O; g, S_I)$ and noting that $p(Y_O|a_i, g, S_I)$ is not a function of S_O ; $p(S_O|a_i, Y_O; g, S_I)$ can be written as:

$$p(S_O|a_i, Y_O; g, S_I) = \text{CONSTANT} \times \frac{1}{[4\pi^2 g^2 \sigma_d^2]^{N/2}} \exp\left[-\frac{1}{2} \Delta P\right] \quad (\text{A.21})$$

where

$$\Delta P = \frac{1}{\sigma_d^2} \sum_{n=0}^{N-1} \{y(n) - s(n)\}^2 + \frac{1}{g^2} \sum_{n=0}^{N-1} \{s(n) - \hat{a}_i^T s(n-1, n-p)\}^2 \quad (\text{A.22})$$

Therefore, maximizing $p(S_O|a_i, Y_O; g, S_I)$ requires minimizing ΔP , or

$$\frac{\partial \Delta P}{\partial s(i)} = 0, \text{ for all } i=0,1,2,\dots,N-1.$$

Unfortunately, this still requires a solution of N linear equations, which in general will be much larger than P (the order of the LPC predictor). Continuing, ΔP can be rewritten as such;

$$\Delta P = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \beta_{ij} [s(i) - m_i] [s(j) - m_j] + \text{CONSTANT} \quad (\text{A.23})$$

where $[\beta_{ij}]^{-1}$ is a covariance matrix. With this formulation for ΔP , MAP estimation of S_O , based on maximizing the probability density function $p(S_O|a_i, Y_O)$ which is jointly gaussian in Y_O , is equivalent to a minimum mean squared error (MMSE) estimate of S_O . Therefore, as the observation window increases (i.e. as N increases), the procedure for securing a MMSE estimate of $s(n)$ approaches a noncausal Wiener filter. With this, the final implementation of the algorithm is presented in figure 4A. This approach can also be extended to the colored noise case as shown in figure 4A. The noise spectral density must be estimated during non-speech activity, as is the case for noise variance in AWGN. Evaluation of this algorithm for the white gaussian noise and colored noise resulted in improved all-pole parameter estimation for a wide range of SNR (Hansen and Clements[35]).

Step 1: Estimate \mathbf{a}_1 from \mathbf{s}_{O_1} .

Use either: i. the first P values as the initial condition vector

or: ii. always assume $\mathbf{S}_1 = \mathbf{0}^T$.

Step 2: i. Using $\hat{\mathbf{a}}_1$ from above, form the estimated power spectrum of the speech under consideration as:

$$P_S(\omega) = \frac{g^2}{\left| 1 - \sum_{k=1}^P a_k e^{-jk\omega} \right|^2}$$

ii. Calculate the necessary gain term using Parseval's theorem.

iii.a Estimate the degrading white noise variance σ_d^2 from a period of silence.

iii.b Estimate the degrading colored noise spectrum $P_D(\omega)$ from a period of silence closest to the utterance.

iv. Construct the noncausal Wiener filter;

$$\text{a) } H(\omega) = \frac{P_S(\omega)}{P_S(\omega) + \sigma_d^2}$$

$$\text{b) } H(\omega) = \frac{P_S(\omega)}{P_S(\omega) + P_D(\omega)}$$

v. Pass the estimated speech $\hat{\mathbf{s}}_1$ through the filter to produce $\hat{\mathbf{s}}_{1+1}$.

vi. Repeat until some specified error criterion is satisfied,

$$\Delta \epsilon < \text{THRESHOLD.}$$

Figure 4A: All-pole modeling/Wiener filtering approach.

Suboptimal Speech Enhancement Algorithm assuming:

a) a AWGN distortion b) a non-white distortion

(Hansen and Clements [35])

ADAPTIVE NOISE CANCELING

The classical approach to adaptive filtering, based on a least mean-squares (LMS) criteria, was first formulated by Widrow et al [50]. This technique has the advantage of requiring no a priori knowledge of the noise signal. The basic principles of adaptive noise canceling is illustrated in figure 5A.a. The input to the adaptive filter is a noise signal $w_1(n)$ that is highly correlated with the additive disturbance, $w(n)$, but uncorrelated with the speech signal $s(n)$. Normally, $w_1(n)$ can be considered as a reference signal from a second

microphone. The reference is filtered to produce the output signal $\hat{w}(n)$ which is an estimate of the additive noise term $w(n)$. Next, the estimate is subtracted from the noisy input $x(n)$ to give the output $z(n)$. This output is an estimate of $s(n)$, and is also used to control the adaptive filter. If $s(n)$ is uncorrelated with $w_1(n)$ and $w(n)$; and if the adaptive filter is adjusted to give an output with the least possible energy, then $z(n)$ is a best least-squares fit to the input speech signal $s(n)$. The proof of this can be carried out by considering the energy in $z(n)$, given by equation (A.24).

$$E\{z^2(n)\} = E\{s^2(n) + \{w(n) - \hat{w}(n)\}^2 + 2s(n)\{w(n) - \hat{w}(n)\}\} \quad (\text{A.24})$$

It is assumed that the noise terms and signal component $s(n)$ are uncorrelated, therefore the energy of $z(n)$ becomes,

$$E\{z^2(n)\} = E\{s^2(n)\} + E\{w(n) - \hat{w}(n)\}^2. \quad (\text{A.25})$$

Also, the signal energy is fixed for the specific speech frame under consideration, therefore minimizing the output energy results in,

$$\min E\{z^2(n)\} = E\{s^2(n)\} + \min E\{w(n) - \hat{w}(n)\}^2. \quad (\text{A.26})$$

So, when the noise canceling filter is adjusted so that $E\{z^2(n)\}$ is minimized, $E\{w(n) - \hat{w}(n)\}^2$ is also minimized. Therefore, $\hat{w}(n)$ is the best least-squares estimate of the primary noise source $w(n)$. In addition, when $E\{w(n) - \hat{w}(n)\}^2$ is minimized, so is $E\{z(n) - s(n)\}^2$ since,

$$\begin{aligned} z(n) &= \hat{s}(n) \\ \{\hat{s}(n) - s(n)\} &= \{w(n) - \hat{w}(n)\} \end{aligned} \quad (\text{A.27})$$

Therefore, $z(n)$ is also the best least-squares estimate of the input speech signal $s(n)$.

In terms of noise canceling for speech inputs, the success of adaptive filtering is highly dependent on the external reference signal. Specifically, that the reference signal be uncorrelated with the speech signal $s(n)$, and correlated with the additive noise. In the techniques considered thus far, only the distorted speech signal is available. It would appear that adaptive filtering is inappropriate, however it is possible to gain information of the

noise characteristics during silent frames. Some disadvantages of such an approach are that, i) the noise is rarely stationary, ii) finite number of samples are not enough to estimate the noise signal, iii) the silent decision is not error free, iv) and such an approach may not be applicable for quantization noise. Though it may be difficult to generate a noise reference, it is quite easy to form a speech reference. Since voiced speech is quasi-periodic, a speech signal delayed by one or two pitch periods will be highly correlated with the true speech signal and uncorrelated with the noise. Figure 5A.b illustrates how the periodic nature of speech can be taken advantage of for effective noise removal. By minimizing the output energy in $\hat{w}(n)$, the resulting estimate $\hat{s}(n)$, will be the best least-squares fit to $s(n)$. Sambur [45] investigated this approach for additive white noise and quantization noise. The adaptive filter in figure 5A.b represents a finite impulse type. The pitch period was estimated using an average magnitude difference function and nonlinear smoothing. For unvoiced sections, two procedures may be applied. One approach simply passes the noisy unvoiced speech through the system unprocessed, the other keeps the LMS filter response constant and processes the unvoiced speech anyway. The filter coefficients b_i 's are updated on a sample by sample basis using the LMS algorithm formulated by Widrow et al [50]. By letting the coefficient vector at time n be represented as $\mathbf{B}_n = (b_0, b_1, \dots, b_L)$, the coefficients used at time $n+1$ may be represented as

$$\mathbf{B}_{n+1} = \mathbf{B}_n + 2\mu \hat{w}(n) \mathbf{X}_{n-T} \quad (\text{A.28})$$

where $\hat{w}(n) = x(n) - \hat{s}(n)$ and $\mathbf{X}_n = \{x(n), x(n-1), \dots, x(n-L)\}$, and μ is a factor that controls stability and rate of convergence. Widrow et al [50] showed that starting with an arbitrary coefficient vector, this algorithm will converge in the mean and remain stable as long as the following is satisfied,

$$\mu > 0$$

$$\text{but also } \mu < 1 / \{\text{LARGEST EIGENVALUE}\}$$

where largest eigenvalue is taken from the matrix $\mathbf{R} = E[\mathbf{X}_n \mathbf{X}_n^T]$. Sambur's implementation is shown in figure 5A.c Results for additive white noise indicate improved quality in the SNR

range of 0 to 10dB. Improvement in 'granular' quality was observed for quantization noise. Variations of this technique involve how the step size μ is chosen, and the choice of filter length L . Another approach might include speech samples which are forward and backward in time. (i.e. this approach estimates $s(n)$ using $s(n-T)$; an approach might consider $s(n-T)$ and $s(n+T)$ to estimate $s(n)$).

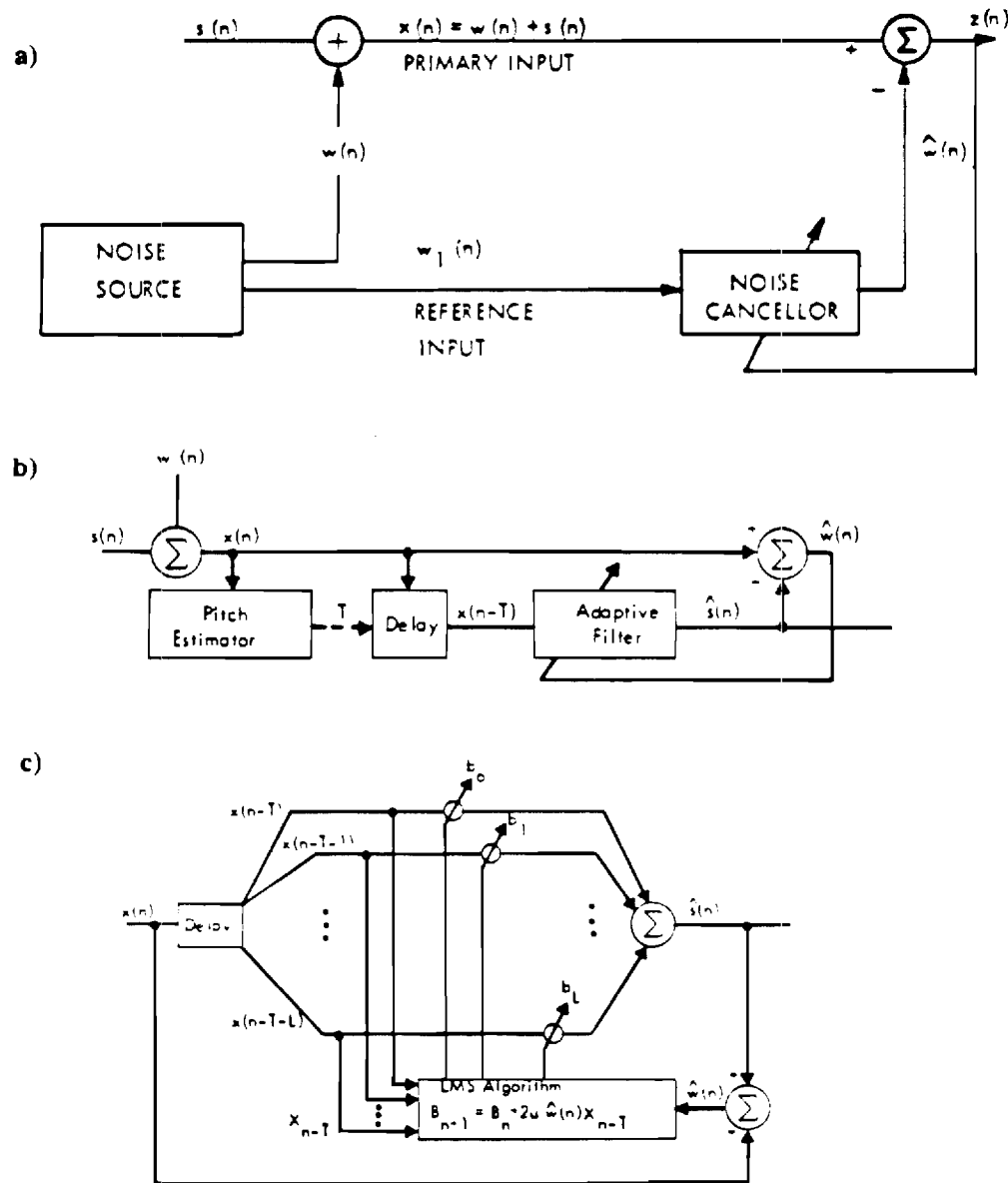


Figure 5A: LMS Adaptive Noise Canceling based on the periodicity of speech.

- a) General noise canceling model (Widrow et al [50]).
- b) Adaptive filtering approach for removing noise from speech.
- c) LMS adaptive algorithm for removing noise from speech (Sarnbur [45]).

COMB FILTERING/ADAPTIVE COMB FILTERING

The last speech enhancement technique to consider is the comb filtering approach. Later improvements have lead to the adaptive comb filtering technique. Comb filtering capitalizes on the observation that voiced segments are periodic, corresponding to the fundamental frequency. The basic process of comb filtering is to build a filter which passes the harmonics of speech, while rejecting frequency components between the harmonics.

The technique is best explained by considering figure 6A. In figure 6A.a, a periodic time waveform is shown. The magnitude spectrum of this waveform is displayed in figure 6A.b. The magnitude spectrum indicates that the energy of the periodic signal is concentrated in small energy bands. It is evident that the magnitude response possesses enough energy to accurately represent the fundamental frequency component. With this, a filter can be implemented which passes the fundamental frequency plus harmonics, while rejecting frequency components between harmonics. Even though speech is only approximately periodic, comb filtering may still be useful in eliminating background noise. An adaptive comb filtering technique introduced by Frazier et al [33], formulates a filter which adjusts itself to both globally and locally to the time varying nature of speech. One disadvantage to comb filtering and adaptive comb filtering, is that it capitalizes on the periodicity of voiced speech. Since the technique is based on the periodicity of voiced speech, only these sections are possible for enhancement. Comb filtering techniques therefore perform poorly for unvoiced speech sections. It is generally known that consonants carry the bulk of linguistic information. Also, vowels (voiced speech) usually possess larger amounts of energy. Therefore, noise degradation tends to mask out unvoiced sections sooner than voiced, thus causing a decrease in intelligibility. Employing a technique which attempts to improve quality in voiced sections, may result in actually decreasing overall speech quality and/or intelligibility.

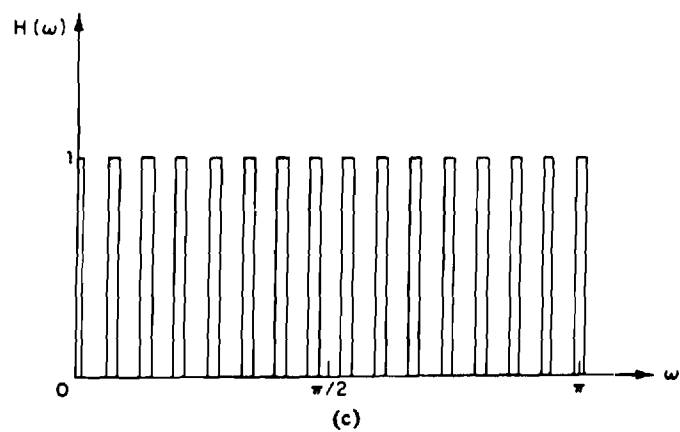
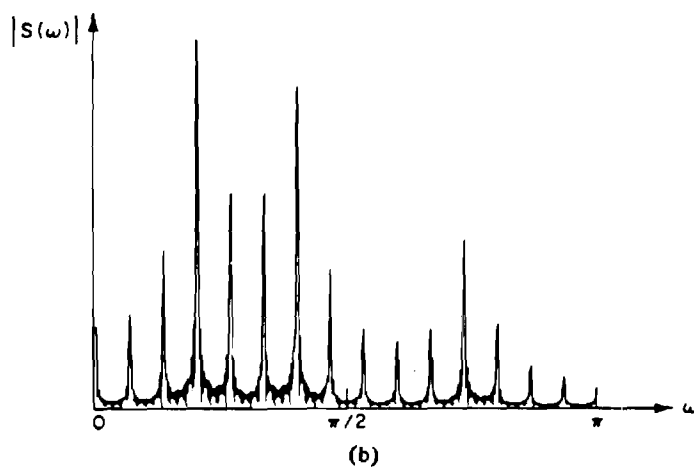
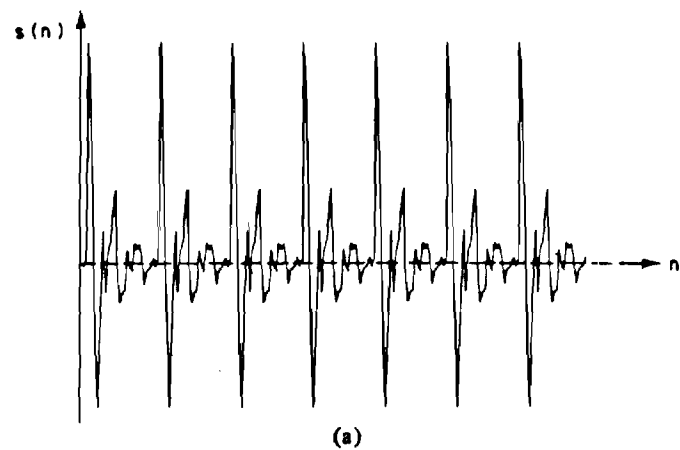


Figure 6A: (a) A periodic time waveform. (b) Magnitude spectrum of the time waveform in (a).
 (c) The frequency response of an ideal comb filter for the time sequence (a).

November 18, 1986

Mr. Dave Kemp
NSA
Mail Stop R 556
Ft. Meade, Maryland 20755

Dear Dave:

Please find attached the progress report for Spring Quarter 1986 for Research Contract
MDA904-85-K-0005: "Research in Digital Speech Processing."

Sincerely,

M. A. Clements

Research in Digital Speech Processing

Progress Report

April - June 1986

Four major components of the project to date have been:

- 1) Investigation of improved enhancement techniques,
- 2) Recognition of components of speech,
- 3) Application of Hidden Markov Modeling, and
- 4) Mixed source speech coding.

I. Enhancement

Existing methods for the enhancement of single channel speech in noise often fall short of attaining their goals, due either to incorrect assumptions or improper goals. The heart of the issue of separating two competing signals occupying roughly the same space in time/frequency lies in applying constraints to the signals. Spectral and correlation subtraction techniques constrain the spectrum of the noise to be constant in time so that on a frame-by-frame basis, the spectrum of the signal can be estimated. The MAP estimation technique discussed by Lim applies the constraint of white noise added to an all-pole speech spectrum, again on a frame-by-frame basis. We have conducted an investigation of both of these techniques and have evaluated them according to criteria other than intelligibility and first impressions. In specific, we have applied objective quality measures based on results obtained in earlier studies. By and large, spectral subtraction only led to negative objective data; i.e., the opposite of enhancement actually takes place. Estimates of the speech parameters using this enhancement were almost always further from correct than with using no enhancement. The only positive feature was the perception of less noise, but the resulting distortion unfortunately more than compensated for this. The MAP technique did demonstrate respectable improvements in objective quality and speech parameter estimation. We adapted the algorithm in a

number of ways, all leading to further improvement. First, the algorithm was generalized to include colored noise. For test data, we used C-130 aircraft noise recorded from mid-fuselage. Second, we enabled re-estimation of the noise spectrum between words. The combination of relaxing the assumptions of whiteness and stationarity of the noise significantly improved both objective quality and speech parameter estimation. Even though the noise sounds fairly stationary, it apparently is not, and our new method helps compensate for this.

II. Recognition of Speech Components

We have started to approach the problem of speech modeling as a system identification problem. If we consider the vocal tract as a time varying linear system with a small number of actual configurations (roughly 64), then the problem can be considered that of estimating the likelihood of each configuration at each point in time given the waveform. The model we employ is a state-space model with observation noise. It has been shown that an efficient method for computing the likelihoods of the competing systems in this framework exists, and is based on Kalman filtering.

We have currently succeeded in adapting this technique to continuous speech. A total of sixty-four competing models are tested in parallel, and the most likely model at each point is selected. This output can be considered similar to that of a vector quantizer, with a new codeword output for each sample. No arbitrary windows are imposed, however, on where the vocal tract configuration can change. Also, different competing models can have different forms, additive noise can be modeled, and efficient computation can be performed. When white noise is used to excite the time varying filters selected in this procedure, intelligible speech results, indicating sufficient modeling precision.

III. Hidden Markov Modeling

Work has continued along the lines indicated in the attached paper from ICASSP-86. The training and transmitter coding issues have basically been resolved. Only fairly simple receiver decoding has been implemented due to computational issues, however. Performance better than a six-bit vector quantizer is obtainable at roughly 2-6 bits per frame, however, using our method. Unfortunately, ten-bit VQ performance is desired. More sophisticated receiver decoding methods should improve things, however.

IV. Coding

Although on this contract no direct student work on mixed-source coding has been funded, we have collaborated on the "self-excited vocoder" work of Rose and Barnwell (attached).

HIDDEN MARKOV MODELS APPLIED TO VERY LOW BIT RATE SPEECH CODING

Eric P. Farges and Mark A. Clements

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia 30332
U.S.A.

Abstract: A new type of very low bit rate speech coder based on a global Discrete Hidden Markov Model (DHMM) of continuous speech for a single speaker is presented here. Several important issues of the training, coding, and decoding procedures are discussed for a 64-state, 1024-observation model. Such a framework is useful in reducing the redundancy in a 10-bit classical Vector Quantizer (VQ), and could lead to a DHMM coder with a bit rate comparable to that of a Segment Vocoder (SV) or a Matrix Quantizer (MQ). This is achieved not only by modelling the long term non-stationarity and the inter-frame time dependencies of the speech, but also by efficiently representing a different kind of information such as vocal tract structure and linguistic patterns.

I. INTRODUCTION

I.1 A new compact and flexible speech model

In recent years, the concept of Hidden Markov Modelling (HMM) of speech has gained considerable popularity due to its successful application in automatic speech recognition. In this paper, we present a new application of such modelling which may be useful to speech coding. In the new model, a 64-state, 1024-observation global Discrete Hidden Markov Model (DHMM) is employed and can be summarized as follows: the system which produces the continuous speech (let us say the human vocal tract) goes through different configurations (called states) as a function of (discrete) time n . A state at time n is a random variable X_n assuming values from a finite state alphabet $S = \{1, 2, \dots, s\}$ ($s=64$). The transitions (or jumps) between the states are probabilistically described by a first order stationary Markov Chain. This in turn is defined by:

- an initial distribution of the states:

$$\pi_0 = (a_i) \quad i=1,s \quad a_i = \text{Prob}(X_1=i) \quad \sum_{i=1}^s a_i = 1$$

- a stochastic transition matrix:

$$\forall n > 1 \quad A = (a_{ij}) \quad i=1,s \quad j=1,s \quad a_{ij} = \text{Prob}(X_n=j / X_{n-1}=i) \\ \sum_{j=1}^s a_{ij} = 1$$

The states (like vocal tract articulatory configurations or linguistic patterns) are not directly observable but are hidden. Nevertheless they manifest themselves through "observations" such as LPC spectra of speech segments. In the discrete HMM case the observations Y_n are assumed to be "drawn" from a finite alphabet $O = \{1, 2, \dots, M\}$ ($M=1024$) made of codewords (indices) for a set of template LPC spectra (all pole models) of a classical VQ codebook C . The speech signal is characterized by a sequence of random variables $Y[1:L] = \{Y_1, Y_2, \dots, Y_L\}$.

The production of an observation Y_n at time n is probabilistically governed by the state the system is in at time n . The production rules are described by a stochastic stationary probability output matrix B :

$$B = (b_{ik}) \quad i=1,s \quad \forall n > 0 \quad b_{ik} = \text{Prob}(Y_n=k / X_n=i) \\ k=1,M \quad \sum_{k=1}^M b_{ik} = 1$$

The sequence of random variables $Y[1:L]$ produced by the underlying state sequence $X[1:L]$ and characterized by the discrete probability mass functions (pmf's) $\{b_i(Y_n)\}$ is called a Probabilistic Function of a Markov Chain. To summarize, a DHMM is uniquely defined by $M = (\pi_0, A, B)$.

I.2 Variations to the basic model

The previous model is referred to as an unconstrained model. A constrained model would force a specific structure on the matrix A . For example if $\forall j > i, a_{ij} = 0$, the model is called *left-to-right*.

A time duration constraint can also be applied to the states. With a DHMM, the probability of staying in state i for N units of time is exponentially decreasing:

$$T(N) = (1 - a_{ii}) a_{ii}^{N-1}$$

which might not be very realistic for speech. A Semi-HMM (SHMM) or jump process introduces a state duration pmf $d_i(n)$ for each state i . Then

$$T(N) = d_i(N)$$

where $d_i(n)$ can be chosen to be some discrete probability mass function. The duration of the system in state i is governed by $d_i(n)$, and the state next visited is governed by the transition matrix A . A SHMM is summarized by (π_0, d, A, B) .

A Continuous HMM (CHMM) uses continuously varying observations not limited to a finite discrete alphabet, but to a continuous support U . Continuous pdf's are used for B :

$$B = (b_i(Y)) \quad i=1,s \quad \text{for } Y \in U.$$

This paper is only concerned with unconstrained DHMM's of speech.

1.3 Motivations for the model

Physical and linguistic reasons: it seems reasonable to consider that the human vocal tract can be represented by a relatively small number of physical configurations (the states), with each configuration being more prone to produce given types of speech spectra (the observations). If one were to assume that speech is composed of a sequence of phonemes, a good description would need to take into account dependencies and relationships between phonemes. In a similar way, the HMM concept statistically identifies time structures and linguistic patterns of the speech and their inter-relationships. Although the underlying states would not represent phonemes, they would exhibit many of the characteristics one would like to see in phonemes, and would enable us to describe speech by a new set of linguistic units. As a result, speech coding could approach bit rates comparable to those of phonemic vocoders, while obviating many of the difficulties. The ultimate bit rate achievable would highly depend on the amount of structure in the underlying model.

Entropy, structure, and bit rate: the entropy of a HMM is defined by [8]:

$$H = - \sum_{i=1}^s \sum_{j=1}^s \pi_i a_{ij} \log_2 a_{ij}$$

where $\pi = (\pi_i)_{i=1,s}$ is the steady state distribution of the states which can be obtained from the transition matrix A by solving the linear system of equations:

$$\begin{cases} \sum_{i=1}^s (a_{ij} - \delta_{ij}) \pi_i = 0 & j = 1, s-1 \\ \sum_{i=1}^s \pi_i = 1 & (\delta_{ij} = 1 \text{ if } i=j, 0 \text{ otherwise}). \end{cases}$$

The entropy represents the average number of bits necessary to encode the states, and is in some sense a measure of the degree of structure (order) of the system: $0 \leq H \leq \log_2 s$:
- if $\pi_i = 1$, $a_{ij} = 1$ and $\forall j \neq i \pi_j = 0$, $a_{ij} = 0$, then $H = 0$: the minimum entropy corresponds to a completely ordered signal with a total a priori information and a known structure.

- if $\forall i, j \pi_i = 1/s$ $a_{ij} = 1/s$ then $H = \log_2 s$ ($H=6$ for 64 states): the maximum entropy corresponds to a completely disordered signal with no a priori information and no underlying structure. Experimental results presented later show a strong underlying structure in the speech, as measured by entropy.

II. THE SPEECH CODER

The very low bit rate speech coder can be divided into 3 distinct procedures (see fig. 1-3):

- *training* in which the best DHMM, $M = (\pi_0, A, B)$ is found given a training sequence of continuous speech.

- *coding* in which the optimum state sequence $X[1:L]$ given the model M and the observed speech $Y[1:L]$ is found, producing a sequence of codewords.

- *decoding* in which the optimum speech sequence $\hat{Y}[1:L]$ given the model M and the transmitted state sequence $X[1:L]$ is found.

Training and coding take place at the transmitter, decoding at the receiver. An optimum bit representation of the states c_n would be transmitted to the receiver, with the number of bits required depending on the entropy of the model. The experimental work completed includes the follow-

ing: A database of 15 minutes of continuous speech from a single male speaker was digitized at 8 kHz, and LPC models were generated for the 60000 15-ms frames of speech. A 10 bit VQ codebook (1024 codewords) was generated with the binary-split K-means algorithm of Buzo et al. [2], and each speech frame was vector quantized. A 64-state, 1024-observation DHMM was trained through the forward-backward algorithm (FBA). The speech was coded with a trellis coding scheme and decoded with a trellis decoding procedure.

II.1 At the transmitter

Training: to estimate the parameters of the model M a maximum likelihood approach can be used through an iterative procedure called the forward-backward algorithm (FBA). Other approaches such as maximum entropy method could probably be used too. The first proof of convergence of the FBA was introduced by Baum et al. [1], then generalized to multivariate continuous distributions by Liporace [4], and extended to the case of multivariate mixture densities by Juang [3]. A practical use of the algorithm was demonstrated for isolated word recognition by Rabiner et al. [5] for small HMM's ($s=5, M=64, L=4000$). For HMM's as large as this study employed, ($s=64, M=1024, L=60000$) significant modification of previously reported FBA implementations was required to alleviate numerical problems. The results of this training will be discussed in section III.

Coding: given the model M and the observed sequence $Y[1:L]$ the goal is to find the sequence of states $X[1:L]$ which is most likely to have produced $Y[1:L]$. A maximum likelihood approach was used in which the sequence $x[1:L]$ which maximizes:

$$P_c(Y[1:L]/x[1:L], M) = \sum_{n=1}^L a_{x_{n-1}x_n} b_{x_n}(Y_n) \quad (a_{x_0x_1} = a_{x_1})$$

was selected. If we define the likelihood function:

$$\mathcal{L}_c = -\log_{10} P_c, \text{ i.e.,}$$

$$\mathcal{L}_c = \sum_{n=1}^L l_{c_n} \text{ where } l_{c_n} = -\log_{10}(a_{x_{n-1}x_n}) - \log_{10}[b_{x_n}(Y_n)]$$

then maximizing P_c is equivalent to minimizing \mathcal{L}_c . A dynamic programming procedure called trellis coding (or Viterbi algorithm) [6] iteratively minimizes the "length" \mathcal{L}_c of the overall state (node) path through a trellis constituted of successive time layers of 64 nodes. The length of a branch between 2 nodes respectively in layers $n-1$ and n is l_{c_n} . The shortest paths ending in each node at every layer n are called "survivors" (there are at most 64 of them). Iterative extensions of the survivors and backtracking lead to the overall best path $X[1:L]$. Some of the trellis coding issues are:

- selection of the initial node (state): The first state can be picked at random, picked according to the steady state likelihood of the states, or picked to minimize the starting lengths. The influence of the starting node is transient, however, and vanishes after a few frames (i.e., survivors starting with different nodes are identical for $n > 5$).

- time constraint: with no time constraint the algorithm could allow as many different states as there are frames per second. It is reasonable, however, to allow no more than 10 (or possibly 20) different states per second. Such a constraint can easily be included in the trellis algorithm through a "sliding window." This feature also makes the coder more robust with respect to noise.

- backward pruning: a full search through the trellis is

not necessary. The extensions at the layer n can be computed from one of up to BP survivors. If no pruning occurs, BP=64. If backward pruning is performed, BP<64 (for example, we could keep only the first 40 survivors). A pruning down to BP=20 generally does not affect the optimality of the results and speeds up the coding algorithm, although even with BP=64, the computations are not burdensome.

- weighting factor α : the likelihood \mathcal{L}_c weights transition and output probabilities equally. Given that the output (observation) sequence is known with certainty, it is reasonable to weight B and A differently using the likelihood:

$$\mathcal{L}_c^* = \sum_{n=1}^L c_n^* : l_{c_n}^* = (\alpha-1) \log_{10}(a_{x_{n-1}x_n}) - \alpha \log_{10}(b_{x_n}(Y_n))$$

II.2 At the receiver

Decoding: given the model M and the state sequence $X[1:L]$ we want to find the most likely sequence of observations $Y[1:L]$. A similar approach to coding is taken. We define:

$$P_d(Y[1:L]/X[1:L], M) = \sum_{n=1}^L [b_{X_n}(y_n)]^{(1-\alpha)} p^\alpha(y_{n-1}/y_n)$$

where $p(y_{n-1}/y_n)$ is the probability of producing observation y_n given that the previous one was y_{n-1} . One alternative for LPC-based models is to use the Itakura-Saito distance measure $D(y_{n-1}/y_n)$ for $\log_{10} p(y_{n-1}/y_n)$, giving a likelihood function:

$$\mathcal{L}_d = \sum_{n=1}^L d_n : l_{d_n} = (\alpha-1) \log_{10}[b_{X_n}(y_n)] - \alpha D(y_{n-1}/y_n)$$

Here, $D(y_{n-1}/y_n)$ is the LPC log-likelihood ratio.

The likelihood is a compromise between most probable observations in a given state ($\alpha=0$) and maximum smoothness of the LPC spectrum ($\alpha=1$). D is referred to as the smoothness function. It can be computed "on line" at the receiver or a distance matrix (stored at the receiver) can be computed at the transmitter. Storage considerations and decoding computation time should indicate what option to use. If no smoothness function is used, decoding is equivalent to selecting the most probable observation in each state, making the coder look like a 64 level VQ (the codewords of the DHMM being more efficient because of increased structure). Some important issues of the trellis decoding algorithm are:

- initial node: the initial node (observation) is known at the transmitter and is transmitted to the receiver, then the decoding can proceed. However, the quality of the decoded speech decreases when time increases since the very long term (200 frames) predictability of the speech is limited.

- observation pegging: to solve the above problem the actual VQ codeword was transmitted to the receiver every P frames (P is the pegging period as well as the trellis depth and decoding delay). A period $P>20$ has only a small influence on the overall bit rate. In the limiting case, $P=1$, the DHMM coder is equivalent to a 1024 level VQ.

- backward pruning: same as coding with BP=100.

- forward pruning: to decrease the decoding time without affecting the optimality of the results, at time n the algorithm looks ahead and selects only the FP most probable observations in state X_{n+1} for which extensions should be computed (need $FP \geq BP$, use $FP=100$).

- weighting factor α : the initial implementation used a constant α . Results indicate that α should be variable:

. in steady state speech regions the smoothness function should be weighted more ($0.5 < \alpha \leq 1$).

. in transient speech regions the output matrix B should

be made adaptable with time $\alpha = \alpha_n$ based on the evolution of the state sequence.

III. PRELIMINARY EXPERIMENTAL RESULTS

The training procedure was run until convergence was approached, with a likelihood value of

$$\log_{10} P = -3.352 \times 10^5$$

Although the model was still improving slightly, the main structures of the matrices were already there (see figures 4-6):

- the initial distribution of the states converged quickly to $a_{ij}=1$ and for $i \neq 19$ $a_i=0$.

- the A matrix displayed, as expected, a strong diagonal structure (see figure 6).

- 56.4% of the entries in A and 89.3% in B were less than 10^{-3} (see figures 4,5).

For a classical VQ, a (non hidden) Markov model was generated by defining the codewords as the states (i.e., states were observations) and a transition matrix was computed by frequency counts on the 15 minutes of speech. The resulting entropy for a 6-bit codebook (64 codewords) was found to be $H = 3.9$. The transition matrix of the 64 state HMM computed on the same data had an entropy of 2.6, which suggests that an inherent underlying structure exists in speech which is not taken into account by the 6-bit VQ. Such techniques as Segment Vocoding and Matrix Quantization capture part of this structure but not all [7].

Preliminary experiments, using the simplest of the coding-decoding techniques described, produced speech superior in quality to that of a 6-bit vector quantizer, but inferior to that of a 10-bit VQ. The work, at this point in time, is very encouraging, considering the large number of unexplored possibilities available.

Conclusion:

A statistical derivation of a new type of speech model has been presented: a global 64 state, 1024 observation DHMM of continuous speech has been proven to be practical, compact, and general, but flexible, automatically trainable, and bit rate efficient (as low as 2.6 bits / frame).

A complex but important underlying structure has been brought to light. Although initially speaker dependent, this new speech model could become speaker independent when plausible linguistic units represented by the states and derived from the strong structure of the model are identified.

Refinements in the modelling and coding-decoding process should produce good speech quality for a very low bit rate coder. Among the many other applications of a DHMM of speech, the detected underlying state sequence could be used for continuous speech recognition.

References:

- [1] L.E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, 41, pp. 164-71, 1970.
- [2] A. Buzo, A.H. Gray, R.M. Gray, J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-28, No.5, pp.562-74, October 1980.
- [3] B.H. Juang, "Maximum-likelihood estimation for mixture multivariate stochastic observations of Markov chains," *AT&T Technical Journal*, 64, No.6, July-August 1985.

[4] L.R. Liporace, "Maximum likelihood estimation for multivariate observations of Markov sources," *IEEE Trans. Information Theory*, IT-28, No.5, pp.729-34, September 1982.

[5] L.R. Rabiner, S.E. Levinson, and M.M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent, isolated word recognition," *Bell System Tech. J.*, 62, No.4, pp.1075-1105, April 1983.

[6] G.D. Forney, JR., "The Viterbi Algorithm," *Proc. IEEE*, 63, No.3, pp. 268-78, March 1973.

[7] D.Y. Wong, "Vector/Matrix quantization for narrow bandwidth digital speech compression," *Signal Technology Technical Report RADC-TR-82-246*, September 1982.

[8] S. Roucos, J. Makhoul, R. Schwartz, "A variable-order Markov chain for coding of speech spectra," *Proc. ICASSP*, Paris, pp.582-5, 1982.

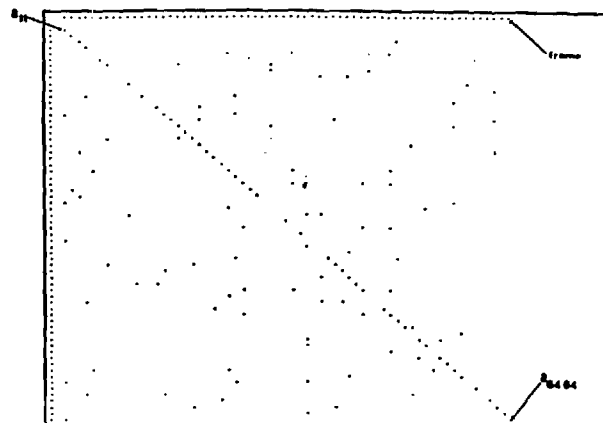


Figure 6: Entries (dots) in transition matrix A greater than 0.1.

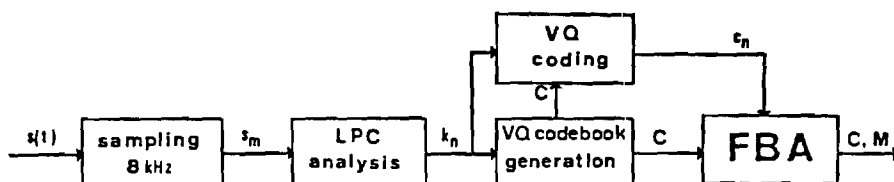


Figure 1: Training of the DHMM at the transmitter.

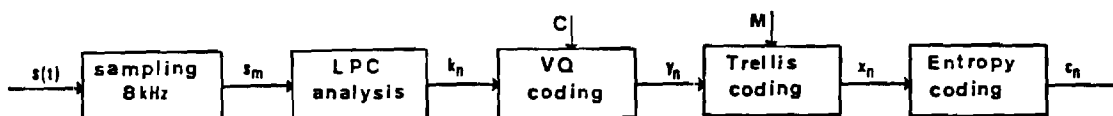


Figure 2: DHMM coding at the transmitter.

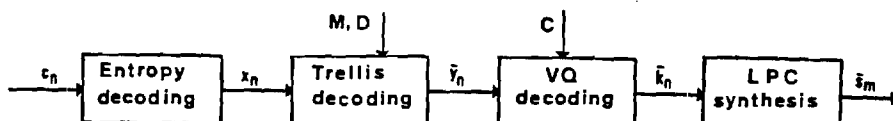


Figure 3: DHMM decoding at the receiver.

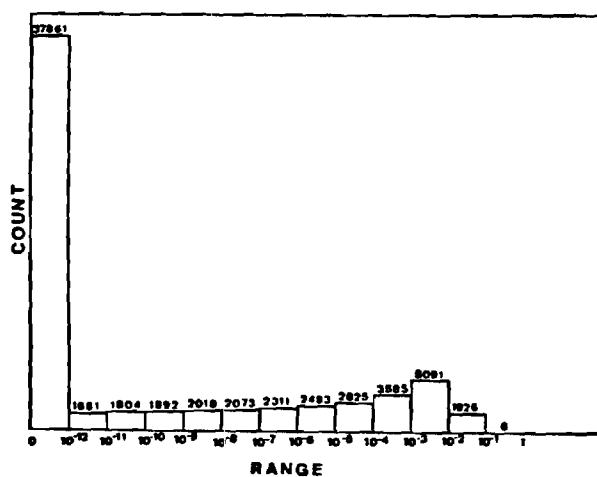


Figure 4: Histogram of entries of output probability matrix B.

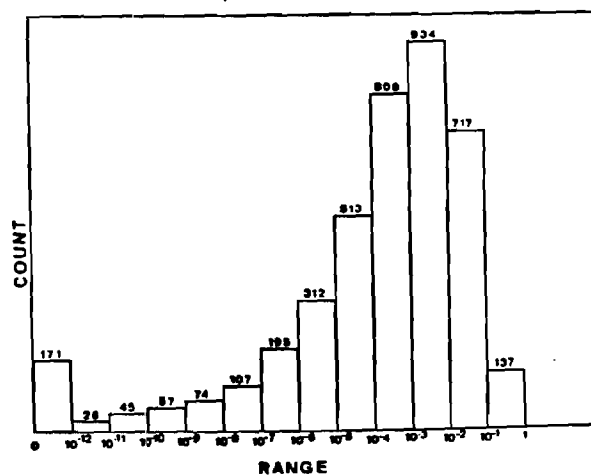


Figure 5: Histogram of entries of transition matrix A.

THE SELF EXCITED VOCODER - AN ALTERNATE

APPROACH TO TOLL QUALITY AT 4800 bps

Richard C. Rose and Thomas P. Barnwell III
Digital Signal Processing Laboratory
Georgia Institute of Technology
Atlanta, GA 30332, U.S.A.

ABSTRACT

This paper presents a new speech coding technique for near toll-quality performance at around 4800 bps. Currently, the most promising techniques at these bit rates emphasize the efficient coding of residual information in a Linear Predictive Coding context. The Multipulse Linear Predictive Coder (MPLPC) [1] and the Code Excited Linear Predictive Coder (CELPC) [2] are examples where considerable success has been achieved at medium bit rates. This paper introduces the Self Excited Vocoder (SEV), a new speech coding technique which belongs to the same class of coders as the MPLPC and the CELPC.

INTRODUCTION

There are many similarities between CELPC and MPLPC. In systems which use a long term predictor, and hence removes the pitch dependent structure from the residual signal, the two techniques differ only in the functions used in coding the source excitations. In particular, MPLPC builds the coded source excitation as a sum of weighted impulse functions. CELPC generates the coded source excitation by searching through a codebook of precomputed sample sequences whose statistics are representative of the whitened residual signal. In both approaches, the optimum excitation is obtained by minimizing the same weighted error function. As will be illustrated in this paper, these are simply special cases of the same general technique.

In terms of the system of Fig. 1, the Self Excited Vocoder may have more than one long term predictor, and has no excitation function, i.e. $V_y(z)=0$. Hence, in its purest form, the SEV, after initialization, applies no bits to coding the residual at all. At first glance, this may seem to be impossible, but a close examination of the system shows that the SEV is actually deriving the source excitation from the past history of the coded excitation function itself using the second long term predictor. In obtaining the optimum source excitation in this way, the SEV minimizes the same weighted error as does the MPLPC and CELPC. Hence, the SEV is another special case of this general class of speech coders. A surprising result is that the SEV is often able to derive enough information from the redundancy of past speech so that no new input is needed to generate the coded source excitation.

This paper has two goals. The first is to present a general formulation which illustrates the relationship between the MPLPC, the CELPC, and the SEV. The second goal is to discuss the operation of the SEV itself.

THE VOCODER MODEL

In linear predictive coders, the spectral envelope and source excitation are modeled separately. The spectral envelope is modeled using a variable predictor, and the prediction residual is modeled by some instantaneous or block quantization technique. This same approach is used here by dividing the model into a short term predictor to represent the spectral envelope and any of three block coding techniques to represent the residual or source. An important point is that any structure or redundancy in the signal that is not modeled by the short term predictor is considered to be part of the source. Hence, the long term predictors in the system of Fig. 1 is considered as part of the excitation function.

Short Term Predictor

The short term predictor model is shown in Fig. 2. The short term predictor $A(z)$ is defined by its predictor coefficients and has predictor delays that are on the order of a few sample periods. In our experimental work, the speech is sampled at 8KHz, preemphasized, and the 10th order predictor is determined over a windowed analysis frame of length 20 msec. The predictor coefficients themselves are computed by solving the standard set of normal equations obtained from the autocorrelation method of LPC analysis [3]. It is important that this analysis is performed over several pitch periods so that a stationary estimate of the spectral envelope is obtained over successive analysis frames. Our spectral model is that of the form

$$A(z) = \sum_{i=1}^P a(i)z^{-i}.$$

The infinite impulse response of the recursive synthesis filter shown in Fig. 2 is denoted as $h(n)$. The synthetic speech $\hat{s}(n)$ is simply the model response to source input $e(n)$

$$\hat{s}(n) = \sum_{i=1}^L h(i)e(n-i).$$

Source

Included in the source is any residual structure that is not represented by the spectral model. These are generally phenomena that must be narrowly resolved in time. They include pitch dependent or long term dependent structure that exhibits significant correlation over delays much greater than the spectral model order. Also included is structure that is random in that it cannot be efficiently modeled by deterministic means. The general model for the coded source is given by

$$e(n) = \sum_{i=1}^M \beta_i \hat{f}_i(n)$$

where $f_i(n)$ is a generalized excitation function depending on the given technique and β_i is the gain associated with that function. Each function is defined over an interval from $n = 0, \dots, N-1$. For MPLPC, the excitation function is a delayed impulse

$$f_i(n) = \delta(n - \gamma_i).$$

For CELPC, $M = 1$ and the excitation function is a sample function from a Gaussian random codebook

$$f_i(n) = v_{\gamma_i}(n).$$

For the SEV, the function is a delayed version of the past excitation. Hence, it represents the response of a long term predictor

$$f_i(n) = e(n - \gamma_i).$$

where in this case γ_i is the predictor delay in samples and is in the range 5-20 msec. This corresponds to the expected range of the pitch period in speech.

The source can consist of any combination of the three techniques. For example, the source excitation $e(n)$ could be given by

$$e(n) = \beta_1 \delta(n - \gamma_1) + \beta_2 v_{\gamma_2}(n) + \beta_3 e(n - \gamma_3).$$

This is in general a recursive expression for $e(n)$. However, if the analysis frame length used for computing source excitation is made short with respect to the long term predictor delay γ_3 , $e(n - \gamma_3)$ will not depend on input during the present frame. It is for this reason that we use an analysis frame length of 5 milliseconds for computing source information. $\hat{s}(n)$ is then given by

$$\hat{s}(n) = \beta_1 h(n - \gamma_1) + \beta_2 \sum_{i=0}^L v_{\gamma_2}(n - i) + \beta_3 \sum_{i=0}^L h(i) e(n - i - \gamma_3). \quad (1)$$

The optimum β_i and γ_i for each excitation function can then be found independently, with the system response due to each added to form the synthetic speech signal. In the next section, a unified formulation for finding all of the three excitation functions is presented.

Long Term Predictor

In voiced speech, successive pitch periods show considerable similarity. This is reflected by a high degree of correlation in the coded source excitation signal at a correlation lag corresponding to a pitch period. This similarity can be represented by a long term predictor whose predictor delay is on the order of a pitch period. Furthermore, we have found that there exists considerable redundancy in speech over time delays that are not directly attributable to pitch. The SEV does not attempt to make any decisions concerning voicing or actual pitch period. It simply searches through the coded source excitation in past analysis frames, looking for an excitation sequence that produces synthetic speech that is highly correlated with that of the original speech. Fig. 3 is an example of the system response to the long term predictor alone. The top waveform represents the original signal, while waveform b represents the weighted response to the source excitation occurring in the previous frames. This corresponds to the last term in Eq. 1, and shows that a significant amount of information can be obtained from the past history of the speech using the long term predictor.

To illustrate the effects of a long term predictor, it is informative to look at the characteristics of a first order long term predictor of the form

$$H(z) = \beta z^{-\gamma}$$

where the delay γ is on the order of 5-20 msec. The impulse response of the system over infinite time is given by

$$h(n) = \delta(n) + \beta \delta(n - \gamma) + \beta^2 \delta(n - 2\gamma) + \dots$$

with associated magnitude frequency response

$$|H(e^{j\omega})|^2 = \frac{1}{1 - 2\beta \cos \omega \gamma + \beta^2}$$

The filter response is peaked at multiples of the frequency corresponding to the predictor delay. However, if the predictor is only allowed to recurse one time as is the case here, the response is of the form

$$|H(e^{j\omega})|^2 = \frac{1 - 2\beta^2 \cos 2\omega \gamma + \beta^4}{1 - 2\beta \cos \omega \gamma + \beta^2}$$

The important point here is that these systems are dominated by the framed nature of the analysis process, and one must be very careful about applying time-invariant interpretations.

The long term predictor is most effective for steady state signals where there is a high degree of interframe redundancy in the speech signal. It does not perform well for transient regions where assumptions of wide sense stationarity are not accurate. Waveform d in Fig. 3 shows the weighted signal to noise ratio computed for the long term predictor alone. Note the minimum signal to noise ratio occurs in the transient region. It is in these regions that pulse excitation provides a more effective means of coding the regular structure of speech. While pulse excitation also does very well at coding more random structure in speech, many additional pulses are needed to do so [4]. Due to the requirements for coding efficiency, codebook excitation is used to code that random structure that cannot be efficiently coded otherwise.

Codebook Excitation

The CELPC falls under the general heading of a stochastic quantizer. Both the transmitter and receiver have identical copies of the same codebook whose entries are populated by precomputed sample sequences, each being 40 points in length. The identifier associated with the codebook entry containing the optimum sample sequence is transmitted to the receiver. After removing the regular deterministic structure from the residual signal, the sample statistics of the residual have been found to approximate a Gaussian distribution [2]. In this case the random codebook can be populated by sample sequences of a Gaussian random process.

While the transmission rate increases with the log of the codebook size, the computation required for the exhaustive search of the codebook increases linearly. Some researchers are presently working on reducing the computation required for an exhaustive codebook search by introducing algebraic structure in the codebook [5]. We have found that the size of the codebook becomes less critical as more structure is removed from the residual signal by way of the long term predictor and pulse excitation. This is essentially an alternative to imposing further structure in the codebook sequences themselves. One must then resolve the tradeoff between the computationally intense burden of an exhaustive codebook search, and the increased bit rate resulting from more accurate deterministic source coding.

Error Weighting Filter

In determining the coded excitation source at low bit rates, the magnitude of the error spectrum can be very large with respect to that of the spectral envelope of the speech signal. It has been found by Atal and Schroeder that weighting this error signal with a weighting filter $W(z)$ will significantly improve the performance of the source coder [2]. It is important that the filter as shown in Fig. 2 is dependent on the short time spectral envelope of the speech. Their weighting filter has the effect of concentrating noise energy in the formant regions of the spectral envelope. The filter is given by

$$W(z) = \frac{1 - A(\alpha z)}{1 - A(z)}$$

where α is a constant between 0 and 1, and effects the amount of energy concentrated in the formant regions. Informal listening tests have shown the presence of some noise weighting filter to be very important in the source coding process.

DETERMINING THE SOURCE EXCITATION

While each of the three source coding techniques best model different types of structure that may exist in the residual, the formulation for determining the optimum excitation function for each is the same. The only difference is the function space from which an optimal excitation function can be chosen. In CELPC, a sample space of random functions corresponding to the 40 point Gaussian random sequences contained in the codebook is searched. In MPLPC, one searches in time over the present analysis frame through a set of delayed impulse functions. The SEV requires searching in time over the range of long term predictor delays through the past history of the source excitation function. For a given technique, the criterion for finding the optimum function is the same. We wish to find the optimum γ and associated β so that $\beta(\gamma)f(n)$ produces synthetic speech that minimizes the energy in the weighted error $W(z)D(z)$ shown in Fig. 2. This is equivalent to coding the inverse weighted speech according to a uniform error criteria given by

$$D(z) = \frac{1}{W(z)} [S(z) - \hat{S}(z)]$$

If γ is held fixed, then it remains only to find $\beta(\gamma)$ in the equivalent expression for $D(z)$ given by

$$D(z) = Y(z) - \beta(\gamma) \frac{F(z)}{1 - A(\alpha z)} \quad (2)$$

$Y(z) = S(z)/W(z)$ is the z transform of the weighted speech, and $F(z)$ is the z transform of the excitation function $f(n)$. We can find $\beta(\gamma)$ by minimizing the mean squared error

$$E = \sum_{n=0}^{N-1} d(n)^2 = \sum_{n=0}^{N-1} [y(n) - \beta(\gamma) \sum_{i=0}^P h(i)f(n-i)]^2$$

where in this case, $h(n)$ is the impulse response of the recursive filter in Eq. 2. The resulting expression for $\beta(\gamma)$ is given by

$$\beta(\gamma) = \frac{\sum_{n=0}^{N-1} y(n)w_{\gamma}(n)}{\sum_{n=0}^{N-1} w_{\gamma}(n)^2}$$

$$= \frac{R_{\gamma w}(\gamma)}{R_{ww}(0)}$$

where $w_{\gamma}(n)$ is

$$w_{\gamma}(n) = \sum_{i=0}^L h(i)f(n-i)$$

or the weighted system response to the given excitation function. Hence, $\beta(\gamma)$ is the normalized cross correlation between the weighted speech and the weighted response to the given excitation. This is the same expression obtained for the gain of an individual pulse in MPLPC where $w_{\gamma}(n) = h(n-\gamma)$. The associated minimum mean squared error is given by

$$E = \sum_{n=0}^{N-1} y(n)^2 - \beta(\gamma)R_{\gamma w}(\gamma) \quad (3)$$

The minimum mean squared error is achieved when $\beta(\gamma)$ is maximized. We can find the optimum excitation function $f(n)$ simply by maximizing $\beta(\gamma)$ over the allowable range of γ .

The gain we give to a given excitation function depends on the degree to which the response to that excitation function is correlated with the weighted speech. As mentioned in the previous section, the source may be built from more than one excitation function. Assuming for a moment that we restrict the possible functions to a single function space, there is a simple procedure for finding additional functions. Suppose we have determined an initial function which is given by γ_1 with corresponding $\beta_1(\gamma_1)$. This process involves removing the effects of the present excitation function and finding the optimum γ_2 and $\beta_2(\gamma_2)$ for the updated speech $y'(n)$ given by

$$y'(n) = y(n) - \beta_1(\gamma_1)w_{\gamma_1}(n)$$

The updated crosscorrelation function is given by

$$R'_{\gamma w}(\gamma) = R_{\gamma w}(\gamma) - \beta_1(\gamma_1)R_{w w}(\gamma)$$

and the expression for $\beta_2(\gamma_2)$ is the same as that given above with the updated crosscorrelation function. Waveform c in Fig. 3 shows the updated signal $y'(n)$ after removing the effects of $w_{\gamma_1}(n)$ due to the long term predictor given in b, from the original $y(n)$ given in a.

It was also suggested in the previous section that each technique best models different types of structure in the residual. In order to take advantage of this, we must be able to measure the performance of each technique on a frame by frame basis. Although it is extremely difficult in general to produce a meaningful measure of subjective performance, we have as a result of our formulation, the noise weighted signal to noise ratio given by

$$SNR = \frac{\sum_{n=0}^{N-1} y(n)^2}{E}$$

E is the minimum mean squared error given in Eq. 3, and depends explicitly on the gain $\beta(\gamma)$. In Fig. 3, waveform d shows the

noise weighted signal to noise ratio for the synthetic speech produced by the long term predictor alone. As mentioned before, it performs poorly in the transition region which is clearly reflected by the SNR contour. This suggests the use of the SNR in determining when additional source excitation functions need be added in building the composite coded source excitation. An important point to note here is that an inaccurate estimation of the SNR threshold results in only a slight increase in transmission rate or slight decrease in SNR.

RESULTS

Clearly, the formulation above can be used to generate a very large family of vocoders which operate over a correspondingly large range of bit rates. The ones of most interest for this paper are the pure SEV's, which, after initialization, operate without any excitation, and which use either one or two long term predictors. Both of these vocoders make no "hard" decisions, such as a pitch period estimation or a voiced/unvoiced decision. The parameters are always chosen simply to minimize the distortion criterion. The effect is that "wrong" decisions, i.e. decisions which disagree with intuition, have only a minor impact on perceived quality.

For the SEV with only a single long term predictor, the excitation bit rate was about 2000 bps. This vocoder resulted in a perceived quality very similar to a 2400 bps fixed-rate pitch-excited vocoder. For the SEV with two predictors, the excitation bit rate was about 4000 bps. The perceived quality was comparable to that of a CELPC with a 1024 coding ensemble.

REFERENCES

- [1] B. S. Atal and J. R. Remde, "A New Model of LPC Excitation for Producing Natural Sounding Speech at Low Bit Rates", *Proc. Inter. Conf. on ASSP*, pp. 614-617, April 1982.
- [2] B. S. Atal, "Predictive Coding of Speech at Low Bit Rates", *IEEE Trans. Communications*, vol. COM-30, pp. 600-614.
- [3] J. P. Markel and A. H. Gray, *Linear Prediction of Speech*. New York, N. Y.; Springer-Verlag, 1976.
- [4] S. Singhal and B. S. Atal, "Improving Performance of Multi-Pulse LPC Coders at Low Bit Rates", *Proc. Inter. Conf. on ASSP*, pp. 1.3.1-1.3.4, April 1984.
- [5] M.R. Schroeder, "Linear Predictive Coding of Speech: Reviews and Current Directions", *IEEE Communications Magazine*, vol. 23, pp. 54-61.

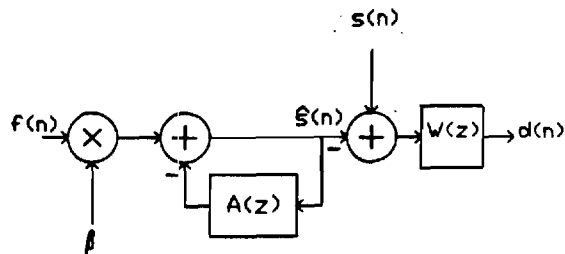


figure 2: Block diagram illustrating procedure for finding generalized excitation function.

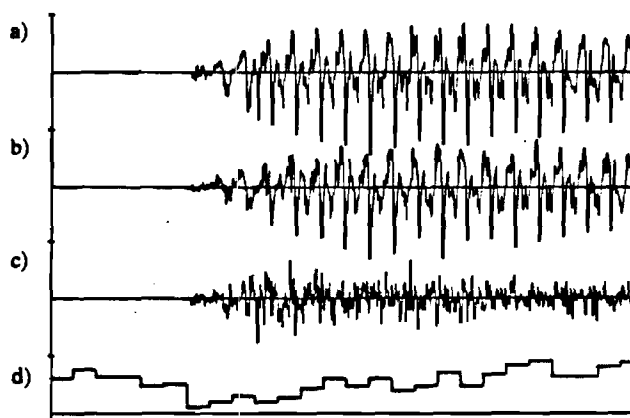


figure 3: a) Noise weighted original speech. b) Response to long term predictor alone. c) Updated speech (error signal). d) noise weighted signal to noise ratio (25 dB full scale).

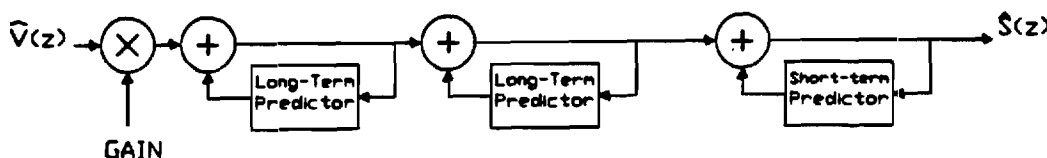


figure 1: Vocoder Model

JUNE 1986

Contract MDA 904-85-K-0005

	Budget	Expended	Balance
Personal Services	27,692	21,825	5,867
Fringe Benefits	1,846	1,717	129
Materials and Supplies	1,000	840	160
Travel	2,000	0	2,000
Total Direct	32,538	24,381	8,157
Overhead	20,661	15,483	5,179
TOTAL	53,199	39,864	13,335

Georgia Institute of Technology

College of Engineering
School of Electrical Engineering
Digital Signal Processing Laboratory
Atlanta, Georgia 30332



GEORGIA TECH 1885-1985

DESIGNING TOMORROW TODAY

November 18, 1986

Mr. Dave Kemp
NSA
Mail Stop R 556
Ft. Meade, Maryland 20755

Dear Dave:

Please find attached the progress report for Summer Quarter 1986 for Research Contract

MDA904-85-K-0005: "Research in Digital Speech Processing."

Sincerely,

M. A. Clements

Research in Digital Speech Processing

Progress Report

June - September 1986

Four major components of the project to date have been:

- 1) Investigation of improved enhancement techniques,
- 2) Recognition of components of speech,
- 3) Application of Hidden Markov Modeling, and
- 4) Mixed source speech coding.

I. Enhancement

We have continued to work with improved versions of the MAP estimator of speech in noise. Our focus recently has been the introduction of time constraints to the speech parameters through use of the spectral-line-pair representation of speech spectra. Although these constraints had originally been used for coding, their power seems to be a result of good speech modeling, which in turn leads to significant improvement for parameter estimation in colored, non-stationary noise. An abstract to ICASSP-87 has been submitted related to these early results.

II. Recognition of Speech Components

We have reformulated the hidden Markov model speech recognition procedure to operate on the Kalman filter outputs. Efficient training and recognition procedures have been formulated. An abstract to ICASSP-87 has been submitted and is attached.

III. Hidden Markov Modeling

Work has continued along the lines proposed in the ICASSP-86 paper by Farges and Clements. Work has focussed on finding the time constraints most appropriate for the receiver decoding. Itakura-type distances for time constraint has not demonstrated adequate performance. Our current theories as to what will work require either storage of a 1024×1024 real matrix, or inordinate amounts of computation. Since the computer in

the Digital Signal Processing Laboratory cannot handle arrays so large, the programs are being transferred and adapted to a larger computer on campus, a Cyber 990.

IV. Coding

We have continued to collaborate with Rose and Barnwell in the self-excited, mixed source vocoder.

Iterative Speech Enhancement With Spectral Constraints

by

John H. Hansen
Mark A. Clements

School of Electrical Engineering
Georgia Institute of Technology
Atlanta, GA 30332

ABSTRACT

A well known speech enhancement technique originally developed by Lim and Oppenheim [1] solves for the maximum likelihood estimate of a speech waveform in additive noise using the constraint that the signal is an all-pole process. Crucial to the success of this approach is the accuracy of the estimates of the all-pole speech parameters at each iteration. An additional disadvantage is that this approach is performed on a single frame basis. Although successful in a mathematical sense, this technique has received little application due to several factors. First, it is an iterative scheme with sizable computational requirements as opposed to a direct form such as spectral subtraction. Second, although the original sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, heuristic convergence criteria had to be employed. In an earlier investigation [2], it was noted that as additional iterations are performed, individual formants of the speech decrease in bandwidth (see fig.1), resulting in unnatural sounding speech. Also, no explicit frame-to-frame constraints are employed. In earlier work on very-low bit rate speech coding, Crosmer [3] was able to encode LPC parameters efficiently by applying the line spectral pair (LSP) transformation. The success of this scheme was a result of some particularly nice properties LSP's were observed to have for speech. First, it was noted that the parameters varied smoothly from frame to frame. Second, it was observed that certain perturbations could be made to the LSP's with little perceptual effect. Third, approximate

formant bandwidths can be computed directly from the LSP's themselves, providing an easy mechanism for ensuring no poles are too close to the unit circle. These three properties allow an efficient method for imposing spectral constraints both within, and across frames, eliminating pole jitter and unpleasant resonances. Impositions of these rules has also been successfully applied to correction of LPC-parameter sequences with random bit errors, showing some of the power of the constraints.

The contribution we have made relates to imposing similar constraints in the estimation of the all-pole parameters of speech in noise. The method basically employs the Lim - Oppenheim procedure with LSP constraints applied after each iteration. Also investigated were constraints applied directly to the pole radii and frequencies of the LPC model. In each case, convergence has been observed.

The techniques were tested on speech degraded by additive white and colored Gaussian noise. Noticeable quality improvement could be observed, although no intelligibility testing has been performed. In addition, we computed various distance measures between the enhanced and original speech signals. LPC-based measures such as the Itakura distance or log-area-ratio distance, showed improvement over the Lim-Oppenheim procedure, suggesting that the new method does more than simply make the speech perceptually cleaner.

Results will be presented comparing the effectiveness of this approach versus that of the unconstrained technique as well as that of spectral subtraction.

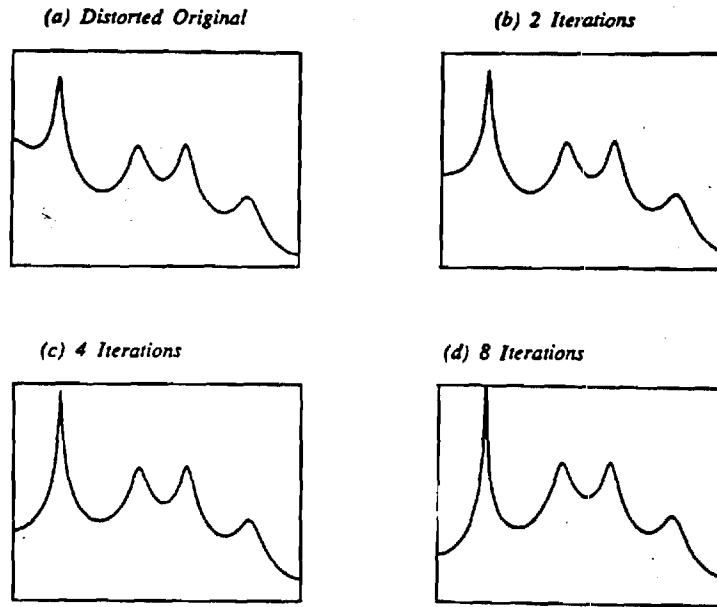


Figure 1: This figure illustrates changes in the vocal tract response caused by the unconstrained enhancement technique. Figures (a) through (e) show how the formant frequencies and associated bandwidths are effected as the iterations increase.

- [1] J.S. Lim, A.V. Oppenheim, " All-Pole Modeling of Degraded Speech," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-26, no. 3, pp. 197-210, 1978.
- [2] J.H. Hansen, M.A. Clements, " Enhancement of Speech Degraded By Non-White Additive Noise," Final Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [3] J.R. Crosmer, " Very Low Bit Rate Speech Coding Using the Line Spectrum Pair Transformation of the LPC Coefficients, " Ph.D. dissertation, School of Electrical Engineerring, Georgia Institute of Technology, Atlanta, June 1985.

HIDDEN MARKOV MODEL SPEECH RECOGNITION BASED ON KALMAN FILTERING

Mark A. Clements
Sungjae Lim

School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Traditional hidden Markov model speech recognition is generally based on a set of parameters (often LPC related) which are extracted at discrete intervals. Such an analysis necessitates use of a discrete-trial hidden Markov model in which the underlying states can only change at intervals related to the frame rate of the analysis. The exact locations of the analysis windows used can influence the front-end outputs and as a result can cause confusion between words differing in short-duration consonants. Our analysis uses a system identification approach which does not require segmentation into frames for front-end processing.

The approach we have adopted is based on a linear model of speech which is time invariant over short intervals. This is the traditional model often used in speech recognition and coding applications. However, we allow for natural, smooth changes occurring in the system as well as additive uncorrelated noise. Our linear model may have periodic excitation (voiced speech) and/or noise input (frication) as well as explicit modeling of time varying system parameters.

Since many phonemes are characterized by a particular evolution in time rather than by steady-state or target spectra, this model is more powerful than more traditional ones. In particular our model is:

$$\mathbf{x}(k) = \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{B}(k)u(k) + \mathbf{G}(k)w(k) \quad (1a)$$

$$y(k) = \mathbf{C}(k) \mathbf{x}(k) + v(k) \quad (1b)$$

where the vector $\mathbf{x}(k) = [x(k), x(k-1), \dots, x(k-p+1)]^T$, $x(k)$ is the speech without noise, $u(k)$ is the periodic input and $\mathbf{B}(k)$ its gain, $w(k)$ the noise input and $\mathbf{G}(k)$ its gain, $\mathbf{C}(k)=[1, 0, 0, \dots]$, $v(k)$ is additive noise, and $\mathbf{A}(k)$ characterizes the time varying vocal-tract filter.

Systems similar to this have been used to model many varied signals arising in sonar, heart monitoring, aircraft control, etc. Recursive linear least squares estimation based on such models falls within general area of Kalman filtering, which allows one to efficiently compute the least squares estimate of $\mathbf{x}(k)$ from the least squares estimate of $\mathbf{x}(k-1)$ and $y(k)$. One crucial step in this technique is to predict as well as possible $y(k)$ based on $y(k-1), \dots, y(0)$. The property we wish to exploit is that if we have modeled the system correctly, the prediction error, $v(k)$, would be white. This idea can be applied to system identification by observing the signals $v_i(k)$ for different competing system models where i denotes the model. A likelihood measure, $p_i(k)$, can be computed for each model based on $v_i(k)$ in an efficient, recursive manner. In particular Lainiotis and Park (1) have shown

$$p_i(k+1) = \frac{N(v_i(k), V_i(k))p_i(k)}{\sum_j N(v_j(k), V_j(k))p_j(k)} \quad (2)$$

where $V_i(k)$ is the expected variance of $v_i(k)$ if model i were correct, and $N(a,b)$ represents the Gaussian density of mean zero and variance b evaluated at a . (Computation of $V_i(k)$ can itself be done recursively and off-line). Although it has been applied with success to cardiac arrhythmia detection (2) and stochastic aircraft control, (3) this technique has not been explicitly applied to speech recognition. In certain special cases, some current speech analysis techniques can be interpreted in this light. For example, the linear predictive model results if only one excitation source exists, system parameters remain constant over short intervals, only abrupt system changes are allowed, and no additive noise exists. Also, recognition based on linear prediction often uses the Itakura distance for segment comparison, which is in some ways similar to the probability calculation in equation (2). It requires, however, that competing models be constant and have exactly the same structure, with only the A matrices varying.

We have currently succeeded in adapting equation (2) to continuous speech. A total sixty-four competing models are tested in parallel, with the most likely model at each sample point selected. This output of this analysis can be considered similar to that of a vector quantizer, with a new codeword output for each sample. We are therefore not bound by arbitrarily placed windows as to where the vocal tract function can change. Also, the competing models can have different forms, additive noise can be modeled, and efficient competition via equation (2) can be performed.

We have also adapted the above formalism to recognition using a hidden Markov model (HMM). The conventional HMM method segments speech into frames and performs vector quantization prior to recognition. Implicit in this technique is the modeling of speech as a discrete-state, discrete-trial Markov process. Our new system would allow a discrete-state, "continuous" trial Markov model. At first glance, one might assume that orders of magnitude more computation would be necessary. This is not the case, however, as we have cast the continuous trial HMM recognition task into a series of matrix operations. Due to the fact that the output of the maximum likelihood selection procedure tends to be relatively stable over significant periods, and since the matrices associated with the HMM's can be diagonalized, we have been able to implement efficient procedures both for training and recognition.

Another variation we have examined relates to using equation (2) again for likelihood calculations, but never making a hard decision as to which model is active. We have shown (on paper) that it is possible to use these likelihoods directly in a HMM. Skipping the intermediate model classification can only improve results, but at a cost of increased computation. We are actively pursuing formulation of algorithms efficient enough to make the above a cost effective procedure.

Our testing of the system on easily confusable (single speaker) words has led to consistent improvement in recognition performance. And we see no reason why improvements for other recognition tasks would not result. In summary, the new model is in many ways a more natural way of analyzing speech for phonetic or word recognition in isolated word environments or in continuous speech.

- [1] Lainiotis, D. G., and Park, S. K., "On joint detection estimation, and system identification discrete data case," *Int. Jour. Control*, vol. 17, no. 3, 1973, pp. 609-633.
- [2] Gustafson, D. E., Willsky, A. S., and Wang, J. Y., "ECG/VCG rythm diagnosis using statistical signal analysis: I. identification of persistent rhythms," *IEEE Trans. Biomed. Eng.*, vol. BME-25, no. 4, July 1978, pp. 344-353.
- [3] Athans, M., Dunn, K. P., Greene, S. C., Lee, W. H., Sandell, N. R., Segall, II, and Willsky, A. S., "The stochastic control of the F-8C aircraft using the multiple model adaptive control (MMAC) method," *Proc. IEEE Decision and Control Conf.*, 10-12, Dec. 1975.

SEPTEMBER 1986

Contract MDA 904-85-K-0005

	Budget	Expended	Balance
Personal Services	27,692	27,277	415
Fringe Benefits	1,846	2,260	(414)
Materials and Supplies	1,000	930	70
Travel	2,000	0	2,000
Total Direct	32,538	30,467	2,071
Overhead	20,661	19,347	1,314
TOTAL	53,199	49,878	3,385

Georgia Institute of Technology

College of Engineering
School of Electrical Engineering
Digital Signal Processing Laboratory
Atlanta, Georgia 30332



GEORGIA TECH 1885-1985

DESIGNING TOMORROW TODAY

April 2, 1987

**Mr. Dave Kemp
NSA
Mail Stop R556
Ft. Meade, Maryland 20755**

Dear Mr. Kemp:

Please find attached the progress report for Fall Quarter Quarter, 1986 for Research Contract MDA904-85-K-0005: "Research in Digital Speech Processing."

Sincerely yours,

**M. A. Clements
Assistant Professor**

MAC:svs

Research in Digital Speech Processing

Progress Report

October - December 1986

Three major components of the project to date have been:

1. Investigation of improved enhancement techniques
2. Recognition of components of speech; and,
3. application of Hidden Markov Modeling.

1 Enhancement

We have continued to work with improved versions for estimation of speech in noise. Work has continued on enhancement of speech in the presence of background noise for improved recognition. The particular method that is currently being explored involves iterative speech enhancement using spectral constraints. Basically, the method forces speech-like characteristics on the enhanced signal. The methods for performing this have been more completely explored now and can be summarized by two procedures. First, time constraints have been applied to the Line-Spectral-Pair (LSP) vocal tract parameters, forcing time evolutions consistent with those observed for natural speech. Second, constraints have been applied across iterations of the algorithm, leading to better behaved convergence characteristics. The description of the work and the major results are summarized in the attached preprint

of the ICASSP-87 paper by Hansen and Clements. We will point out some of the results referring to the figures in the paper.

The theoretical limit of performance uses the true LPC spectrum of the speech for the Wiener filtering step. In a real-life situation, the true LPC spectrum is unknown, since the speech is corrupted by noise. We are adding noise to uncorrupted speech, and hence have direct access to the LPC parameters enabling us to check our results. One measure of how good the enhancement procedure performs is the Itakura-Saito likelihood measure. It is also a fairly relevant measure since it is often used for speech recognition systems. Table 1 shows the results for white noise and 5 dB SNR across a wide range of different speech segments. *Original* means application of the Itakura-Saito measure comparing noisy and undegraded speech; *Lim-Oppenheim* refers to the algorithm using no constraints; *Hansen-Clements* refers to the best of the algorithms currently studied; and *True LPC* refers to the aforementioned theoretical limit. The numbers are in units of distance, with smaller being better and 0 meaning exact spectral match. As can be seen, the constraints considerably improved the enhancement.

Figure 3 summarizes distance across a few of the test systems and SNR's. The clear winner (line e) used LSP constraints across frames (inter-frame) and autocorrelation coefficient constraints across iterations (intra-frame). Figure 4 shows spectral subtraction, unconstrained, inter-frame-constrained, and inter-frame plus intra-frame-constrained across different SNR's. Again, the clear winner over all SNR's used the most constraints.

A measure more relevant to speech quality is the *Klatt* measure, our weighted spectral-slope distance measure. It has shown consistently higher correlation with subjective quality data than have other more traditional measures (e.g., Itakura-Saito). Data identical to that of figures 3 and 4 and table 1 were computed using this measure. Very similar results were obtained for the Klatt measure as the Itakura-Saito measure.

Our continued work is how to apply the constraints to noise which is non- white and only short-term stationary.

2 Recognition of Speech Components

We have been developing and testing a new algorithm which is useful for better recognition of short-duration interior consonants. The technique is based on a procedure which approximates a discrete- state-continuous trial hidden Markov model and uses a bank of Kalman filters as a front-end. Please find the preprint for the ICASSP-87 paper attached. The important result is the reduction in error rate by a factor of six over more traditional models.

3 Hidden Markov Modeling

Computer work has continued on global Hidden Markov Modeling for speech analysis and coding. The computer code has been converted and transferred to a bigger and faster computer with less competition. In addition, a stronger theoretical basis for the work has been formulated. New methods for efficiency improvement (e.g., removing logarithms in many places) have been made possible as a result and have enabled substantial speeding up of the training procedure.

Iterative Speech Enhancement With Spectral Constraints

John H. Hansen and Mark A. Clements

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia 30332

Abstract

A new and improved iterative speech enhancement technique based on spectral constraints is presented in this paper. The iterative technique, originally formulated by Lim and Oppenheim, attempts to solve for the maximum likelihood estimate of a speech waveform in additive white noise. The new approach applies inter- and intra-frame spectral constraints to ensure convergence to reasonable values and hence improve speech quality. An extremely efficient technique for applying these constraints is in the use of line spectral pair (LSP) coefficients. The inter-frame constraints ensures more speech-like formant trajectories than those found in the unconstrained approach. Results from speech degraded by additive white Gaussian noise show noticeable quality improvement.

Introduction

The successfulness of an enhancement algorithm rests on the goals and assumptions used in deriving the approach. Depending on the application, a system may be directed at one or more objectives such as improving overall quality, increasing intelligibility, reducing listener fatigue, etc. Three assumptions normally made include: i) that the noise distortion be additive, ii) that only the degraded speech signal is available, and iii) that the noise and speech signals are uncorrelated. In general, constraints placed on the speech model improve the potential for separating speech from background noise. However, such systems are also more sensitive to "deviations" from these constraints. The degradation considered is additive white Gaussian noise. The basis of the technique is an iterative enhancement approach based on noncausal Wiener filtering originally formulated by Lim and Oppenheim [1]. This approach attempts to solve for the maximum likelihood estimate of a speech waveform in additive white noise using the constraint that the signal is an all-pole process. Crucial to the success of this approach is the accuracy of the estimates of the all-pole speech parameters at each iteration. One advantage of the Wiener filtering approach is that no "musical tone" artifacts are present after processing as can be observed in spectral subtraction techniques. In addition, under certain conditions, it can be shown that it is the optimal solution in the mean-squared sense for a white noise distortion. Although successful in a mathematical sense, this technique has received little application due to several factors. First, it is an iterative scheme with sizable computational requirements as opposed to a direct form such as spectral subtraction. Second, although the original sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, heuristic convergence criteria had to be employed. After an extensive investigation [2], this approach was found to produce significant levels of enhancement for white Gaussian noise in 3-4 iterations. The technique was generalized to allow for colored aircraft noise. Various spectral estimation techniques were employed for securing estimates of the colored background noise and although the noise was not stationary, estimates were performed prior to application of the algorithm.

With these assumptions, good enhancement took place in 2-3 iterations. It is assumed that in a real-time environment however, noise spectral estimates could be gathered and updated during silent intervals. An important observation which could be made from this previous work was that as additional iterations were performed, individual formants of the speech decreased in bandwidth (see fig.1), resulting in unnatural sounding speech. Frame-to-frame pole jitter was also observed which contributed to unnatural sounding results. Also, the original technique employs no explicit frame-to-frame constraints. Since the original algorithm already constrains the speech to be the response from an all-pole system, applying further constraints on the pole movements may improve the algorithms performance. One set of constraints were applied directly to the LPC poles. These results were quite encouraging, yet computationally intensive. A new approach for implementing the spectral constraints was formed by employing the line spectral pair (LSP) transformation as a method for representing the vocal tract spectrum. This method of specification allowed constraints to be efficiently applied to the speech model pole movements across time (inter-frame) so that formants lay on smooth tracks. In addition, constraints could also be easily applied across iterations (intra-frame) on a frame-by-frame basis.

Iterative Speech Enhancement

Enhancement based on the estimation of all-pole speech parameters in additive white Gaussian noise was investigated by Lim and Oppenheim [1], and later for a colored noise degradation by Hansen and Clements [2]. It was shown that the estimation procedures which result in linear equations without background noise, become nonlinear when noise is introduced. However by allowing a suboptimal procedure, an iterative algorithm results which possesses the property that the estimation procedure is linear at each iteration.

Consider the statistical parameter estimation of speech in the presence of noise. Over a short-time basis, the speech signal can be represented as the following difference equation:

$$s(n) = \mathbf{a}^T \mathbf{s}(n-1, n-p) + g w(n) \quad (1)$$

where $\mathbf{a}^T = [a_1, a_2, \dots, a_p]$ represents the all-pole predictor coefficients. Substituting the degraded speech into the speech model gives the following equation for the observation vector:

$$\mathbf{y}_0 = \mathbf{y}(N-1, 0) = \mathbf{s}(N-1, 0) + \mathbf{d}(N-1, 0) \quad (2)$$

$$\mathbf{y}_0 = \mathbf{a}^T \mathbf{y}(n-1, n-p) + g w(n) + \mathbf{d}(n) - \mathbf{a}^T \mathbf{d}(n-1, n-p)$$

where $\mathbf{s}(N-1, 0)$ are N samples of original speech, and $\mathbf{d}(N-1, 0)$ represents the additive background noise. The $2p + 1$ unknowns include the predictor coefficients \mathbf{a} , initial conditions for the predictor given by $\mathbf{S}_1 = \mathbf{s}(-1, -p)$, and the gain factor g for the input excitation. Consider the case where all unknown parameters are random with a priori Gaussian probability density functions. The basic procedure used is a maximum a priori (MAP) estimator, which maximizes the probability density function of

the parameters given the observations. Therefore, a, g, S_1 are chosen to maximize the probability density function $p(a, g, S_1 | Y_0)$. The procedure requires that a be chosen to maximize $p(a | Y_0)$, noting that the estimate is conditioned on the noisy observations Y_0 . Using Bayes' rule, $p(a | Y_0)$ can be written as a product of terms involving $p(Y_0 | a, g, S_1)$. When the Gaussian density function $p(Y_0 | a, g, S_1)$ is expanded, it can be shown that the mean and variance are functions of the predictor coefficients a . Therefore the resulting equations for maximizing $p(a | Y_0)$ are nonlinear, involving partial derivatives with respect to a . Lim and Oppenheim considered a suboptimal solution employing a two step approach based on MAP estimation of S_0 given Y_0 , followed by MAP estimation of a given \hat{S}_0 , where \hat{S}_0 is the result of the first estimation. Observations indicate that this algorithm converges to a local maximum of the joint density $p(a, S_0 | Y_0, g, S_1)$. In particular, if the probability density function is unimodal, and the initial estimate for a is such that the local maximum equals the global maximum, then the procedure is equivalent to the joint MAP estimate of a and S_0 . After some simplification, the MAP estimation of S_0 , based on maximizing the probability density function $p(S_0 | a, Y_0)$ which is jointly Gaussian in Y_0 , is equivalent to a minimum mean squared error (MMSE) estimate of S_0 . Therefore as the observation window increases in length, the procedure for obtaining a MMSE estimate of $s(n)$ approaches a noncausal Wiener filter. With this, the implementation of the algorithm is presented in Figure 2. This approach can also be extended to the colored noise case as shown. As indicated, the background noise spectral density must be estimated during non-speech activity.

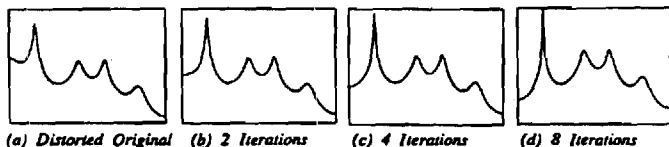


Figure 1: Variation in vocal tract response across iterations.

As indicated, the sequential MAP estimation technique increases the joint likelihood of the speech waveform and all-pole parameters, yet a heuristic convergence criterion had to be employed. Also, as additional iterations were performed, individual formants of the speech decrease in bandwidth as indicated in figure 1. Frame-to-frame pole jitter was also observed. Both effects contributed to unnatural sounding speech. The goal, therefore is to impose constraints on the pole movements across time (inter-frame) and iterations (intra-frame). An initial approach was to limit the poles from moving too close to the unit circle by performing an off-axis spectral evaluation where the z -transform is evaluated on a circle further away from the poles of the spectral model. Other approaches considered included applying constraints directly to the pole radii and/or angular displacements in the LPC model. Performance of such inter and intra-frame constraints lead to encouraging results, but at the expense of a p th order root-solve and a pole ordering step per frame for each iteration. Since root solving is not always numerically accurate and ordering can be inconsistent across frames, a more robust approach was sought to implement these constraints. Previous success of the line spectral pair (LSP) transformation in speech coding by Crosser [3], led to the use of LSP's for this purpose.

Line Spectral Pair Representation of Spectral Characteristics

The LSP transformation may be viewed as an alternative representation of the LPC spectrum. The LSP coefficients are obtained from the LPC prediction coefficients by combining the forward and backward predictor polynomials as follows:

$$P(z) = A(z) + B(z), \quad Q(z) = A(z) - B(z). \quad (3)$$

The vocal tract transfer function is given by $g/A(z)$, and M is the order of the LPC speech model. The resulting polynomials $P(z)$ and $Q(z)$, are symmetric and antisymmetric, respectively, with a root of $P(z)$ at $z=+1$, and a root of $Q(z)$ at $z=-1$. The remainder of the roots of P and Q all lie on the unit circle. Since the roots occur in conjugate pairs, the original polynomial can be represented by M real numbers. The angles of the roots, $\{\omega_i, i=1,2,\dots,M\}$, are called the *line spectrum pairs*.

The LSP's possess several important properties which make them attractive for use in applying spectral constraints. One important characteristic is that if the vocal tract polynomial $A(z)$ has all its roots inside the unit circle (i.e., a stable filter), then the roots of P and Q will be interleaved around the unit circle [3]. If two adjacent LSP frequencies are identical, it indicates that a root of $A(z)$ lies on the unit circle.

In addition to their attractive representation of the LPC spectrum, the LSP coefficients offer the possibility of a more direct representation of perceptually important information. Specifically, there is a firm statistical relationship between the locations and bandwidths of the speech formants and the locations of the roots of P and Q respectively. Since roots of the P polynomial correspond approximately to locations of formant center frequencies (when a formant is present), the P polynomials' LSP coefficients are termed *position coefficients*. It can be shown that the closer two LSP coefficients are together, the narrower the bandwidth of the corresponding pole of the vocal tract filter. Therefore, formants are indicated when two LSP coefficients are close together. When LSP coefficients are far apart, they indicate poles which contribute only to the overall spectral shape. Because of their relationship to the presence or absence of a formant by their nearness to a position coefficient, the coefficients of Q are termed *difference coefficients*. Given the LSP coefficients, the position coefficients are simply the odd index LSP coefficients, $\{p_i = \omega_{2i-1}, i=1,2,\dots,M/2\}$. The difference coefficients are given as follows:

$$\{d_i\} = \min_{j=-1,1} (|\omega_{2i+j} - \omega_{2i}|), \quad i = 1,2,\dots,M/2 \quad (4)$$

where the sign of d_i is positive if ω_{2i} is closer to ω_{2i+1} , and otherwise is negative. With this interpretation, a new enhancement technique based on Wiener filtering is now possible by imposing constraints on the LSP coefficients.

Step 1: Estimate a_1 from $S_{0,1}$.

Use either: i. first P values as the initial condition vector
or: ii. always assume $S_1 = 0$.

Step 2: i. Using a_1 , estimate the speech spectrum:

$$P_g(\omega) = \frac{g^2}{|1 - \sum_{k=1}^p a_k e^{-j\omega k}|^2}$$

ii. Calculate gain term using Parzen's theorem.

iii. Estimate either the degrading

a.) white noise variance σ_a^2 , or b.) colored noise spectrum $P_D(\omega)$ from a period of silence closest to the utterance.

iv. Construct the noncausal Wiener filter;

$$a.) H(\omega) = \frac{P_g(\omega)}{P_g(\omega) + \sigma_a^2} \quad b.) H(\omega) = \frac{P_g(\omega)}{P_g(\omega) + P_D(\omega)}$$

v. Filter the estimated speech \hat{s}_1 to produce \hat{s}_{1+1} .

vi. Repeat until some specified error criterion is satisfied, $\Delta e < \text{THRESHOLD}$.

Figure 2: Enhancement Algorithm based on All-pole modeling/Wiener filtering. a) a AWGN distortion b) a non-white distortion

Enhancement with Spectral Constraints

Consider the statistical parameter estimation of speech in the presence of noise, where all unknown parameters are random with a priori Gaussian probability density functions. It can be shown that MAP estimation of a , g , and S_1 given the noisy observations Y_0 , results in a set of nonlinear equations. Therefore, instead of joint estimation of a and S_0 , a suboptimal solution is formulated employing a two step approach based on

MAP estimation of S_0 given Y_0 , followed by MAP estimation of a given \hat{S}_0 , where \hat{S}_0 is the result of the first estimation. Since speech can be considered short-time stationary, frame-to-frame spectral constraints may aid in enhancement. The new approach imposes such constraints on the vocal tract spectrum between MAP estimation steps. The procedure for obtaining the MAP estimate of a from $\text{MAX } p(a|\hat{S}_0, g, S_1)$ remains the same. The next step is to apply spectral constraints to \hat{a}_1 which will ensure that; i) the all-pole speech model is stable, ii) it possess speech-like characteristics (i.e., poles are not too close to the unit circle causing narrow bandwidths), and iii) the vocal tract characteristics do not vary wildly from frame-to-frame when speech is present. Due to this constrained approach, an improved estimate \hat{a}_1 results. Given this new estimate, the second MAP estimation of S_0 given \hat{a}_1 can be carried out by maximizing $p(S_0|\hat{a}_1, Y_0, g, S_1)$. Since $p(S_0|\hat{a}_1, Y_0, g, S_1)$ is still jointly Gaussian in Y_0 , the resulting MAP estimate is equivalent to a MMSE estimate of S_0 . Again, in the limiting case, the procedure for obtaining the MMSE estimate of $s(n)$ approaches a noncausal Wiener filter. Once this new estimate of $\hat{S}_{0,1}$ is formed, the iterative procedure continues by re-estimating \hat{a}_1 , applying constraints to \hat{a}_1 , and then forming the noncausal filter using \hat{a}_1 to re-estimate $\hat{S}_{0,1}$. This continues until some convergence criterion is satisfied. The procedure for implementing these constraints will now be addressed.

Two classes of spectral constraints are considered; inter-frame (across time), and intra-frame (across iterations). Two approaches are considered: a fixed frame rate, and a variable frame rate approach. In the first of these, the LPC predictor coefficients, a , are first converted to LSP position and difference coefficients. Next, each frame's energy is observed, and if it is above some threshold, it is classified as voiced speech; if it is below, then it is either noise or unvoiced speech. A local running count L_1 is kept for the number of consecutive frames which fall below the energy threshold. If L_1 reaches L_{MAX} , then all subsequent frames below the threshold are classified as noise. This allows for further smoothing for long periods of silence. The position coefficients for each frame are smoothed using a weighted triangular window with a variable base of support (1 to 5 frames). If a frame has been classified as noise, maximum smoothing is performed. In addition, the lower formant frequencies are smoothed over a narrower triangle width than for those position coefficients at higher frequencies. This preserves perceptually important speech characteristics found in the lower formants. No smoothing is performed on the difference coefficients since they are more closely related to formant bandwidth than formant location. However, it is possible that a difference coefficient falls within a "forbidden zone," (i.e., the region within d_{MIN} of a position coefficient). When this occurs, the LPC analysis has most likely overestimated the Q of a particular pole. Since this causes unnatural sounding speech, (as in the unconstrained approach), the value of $|d_i|$ is set to d_{MIN} . Finally, the position and difference coefficients are combined to form the constrained LPC predictor coefficients \hat{a}_1 .

The second inter-frame constraint approach considered is a variable frame rate technique which takes advantage of the interpolation properties of the LSP coefficients. The speech signal is first divided into segments, where segments are chosen such that they are long when the speech spectrum is varying slowly and short when the speech spectrum is varying quickly. The LSP coefficients are reconstructed with linear interpolation used to compute the coefficients for intermediate frames.

The segmentation algorithm begins with a step to determine the onset/offset of speech. This is carried out by thresholding the LPC residual energy, which produces relatively long segments. Next, the long segments are subdivided based on the curvature of the position coefficients. This is performed by computing a gain-normalized Itakura-Saito measure of the spectral distance between the frequency response of two adjacent frames. The procedure continues by computing the distortion of

position coefficients for successively longer segments until the distortion exceeds a threshold T_D . At that point, a subsegment boundary is set, with the intermediate position coefficients reconstructed via linear interpolation. During this step, the length of a subsegment is also limited to L_{MAX} to prevent excessively long segments which might contribute to muffled or unnatural sounding speech. The advantage of this approach is that it incorporates more information from adjacent frames when the spectrum indicates similar characteristics. Yet, it also reduces the effects of adjacent frames when the spectrum is significantly different as in the case of a transition from unvoiced passages to noise. This in effect, distorts the position coefficients as little as possible when associated difference coefficients indicate the presence of formants. Difference coefficients for each frame, (or an average set across a segment) are used to compute the predictor coefficients \hat{a}_1 . The difference coefficients are required to be at least d_{MIN} or greater in distance from adjacent position coefficients to ensure that poles from the LPC filter do not move too close to the unit circle.

Inter-frame constraints are applied to a single frame across iterations, and as such require the frames' previous estimates to be available. The motivation for such constraints is that under certain conditions, pole locations for the same frame vary significantly from their previous estimated values. Since the present estimate of \hat{a}_1 affects the next estimate of $\hat{S}_{0,1}$, sections of $\hat{S}_{0,1}$ will also vary significantly across iterations. In addition, previous results based on objective speech quality measures indicated that the unconstrained approach produced minimum objective measures at different iterations for different classes of speech. For example, maximum overall speech quality was observed for additive white Gaussian noise in three iterations. This was also true for vowels and fricatives. However, glides required two iterations, nasals, liquids, and affricates between five and six. It is therefore desirable to be able to affect the convergence rate so that the best objective measure of quality occurs at the same iteration across all classes of speech. Improved quality as measured by objective measures may also result in improved estimation of \hat{a}_1 . By constraining the vocal tract filter to be a function of its previous estimates, it may be possible to accomplish this. Two approaches are considered, one applied to the autocorrelation lags, the other to the position coefficients. The first approach simply weights the present set of autocorrelation lags with the same frame from previous iterations. This technique is very easy to perform, since the autocorrelation lags must be computed in order to estimate the predictor coefficients a . The second approach weights position coefficients with those from the same frame but previous iteration. If the corresponding difference coefficient indicates the adjacent position coefficient to represent a formant, this approach has the effect of constraining the formants to lie along smooth tracks across iterations.

Results

Speech degraded by additive white Gaussian noise was processed using various configurations of the new constrained enhancement algorithm. Energy thresholds for inter-frame constraints were obtained from frame energy histograms at each signal-to-noise ratio. Excellent enhancement resulted for a wide range of threshold values. Intra-frame constraints were applied across two to three iterations. Informal listening tests indicated noticeable quality improvement, although no intelligibility testing has been performed. However, there has been extensive work carried out in the area of objective speech quality measures [4]. Good correlation has been shown to exist between subjective quality and objective measures. Therefore, objective measures including: the Itakura-Saito likelihood ratio, log area ratio, and weighted spectral slope measure were used for evaluation. Figure 3 illustrates a comparison of

typical results for the various constraint approaches. Itakura-Saito measure is plotted versus signal-to-noise ratio for a white noise distortion. Plot *a* represents the original distorted speech. Plots *b* through *e* represent combinations of inter-frame constraints (both fixed and variable rate), and intra-frame constraints (applied to position coefficients/autocorrelation lags). All configurations examined showed significant improvement in Itakura-Saito measures. Threshold settings for the variable frame rate inter-frame constraint were somewhat sensitive to varying noise levels. However, the fixed frame approach by itself, and with either autocorrelation or position intra-frame constraints gave impressive results with little sensitivity to varying levels of SNR. In order to determine a limit on the level of enhancement, the original undistorted predictor coefficients *a* were used in the unconstrained algorithm. In essence, the two step MAP estimation approach is now reduced to a single MAP estimate of S_0 , and therefore represents the theoretical limit for enhancement using Wiener filtering. Plot *f* indicates this limit. Although only Itakura-Saito measures are shown, similar improvement was also observed for log area ratios and weighted spectral slope measures. Figure 4 compares the new approach to existing techniques. Plot *b* shows results from spectral subtraction as formulated by Boll [5]. An evaluation was performed for both half and full-wave rectification, along with one to five frames of magnitude averaging; where these points represent the best results. Plot *c* is from the unconstrained Wiener filtering technique. Plots *d* and *e* are typical values for the inter-frame constraint (fixed frame rate), and inter plus intra-frame constraints (fixed frame and autocorrelation lags). Again *f* indicates the limit for the Wiener filtering approaches.

Sound Type	Itakura-Saito Likelihood Measure			
	Original	Lim-Oppenheim	Hansen-Clements	True LPC
Silence	1.634	1.649	0.842	0.319
Vowel	4.020	3.299	1.651	0.582
Nasal	19.814	17.656	3.968	0.324
Stop	7.261	3.979	1.099	0.435
Fricative	3.739	3.509	1.766	0.649
Glide	1.525	1.442	1.131	0.705
Liquid	9.597	4.545	0.998	0.303
Affricate	3.924	2.702	2.229	0.323
Voiced + Unvoiced	5.838	4.293	1.761	0.519
Total	4.022	3.151	1.364	0.433

SNR = +5dB

Table 1: Comparison of algorithms over sound types for white Gaussian noise.

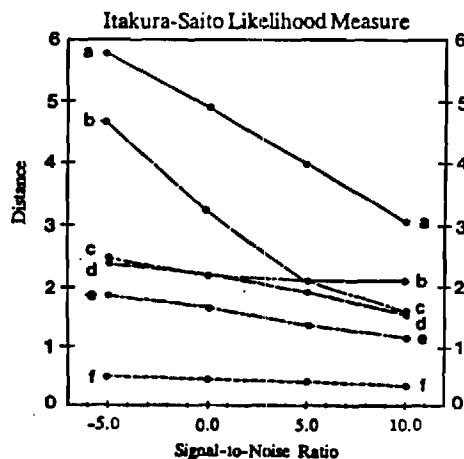


Figure 3: Comparison of constraint algorithms over SNR.

- a.) Original Distorted Speech
- b.) Inter-Frame Constraint: Variable Frame
- c.) Inter-Frame Constraint: Fixed Frame
- d.) Inter & Intra-Frame Constraints: Fixed Frame, Position
- e.) Inter & Intra-Frame Constraints: Fixed Frame, Autocorrelation
- f.) Theoretical limit: using undistorted LPC coefficients, *a*.

Performance evaluation over sound classes was accomplished by hand partitioning speech into segments. Entire sentences were processed, and objective measures from each class were computed. Table 1 summarizes this comparison between the unconstrained Lim-Oppenheim technique to that of the inter and intra-frame constraint approach. Measures for the theoretical limit using undistorted LPC predictor coefficients *a* are also indicated. Improvement is indicated for all types of speech. In addition, the constrained approach produced superior objective measures of quality across all speech classes at the same iteration. These results clearly indicate improvement over the unconstrained approach as well as spectral subtraction for additive white Gaussian noise.

Conclusions

The application of spectral constraints to noncausal Wiener filtering results in improved speech enhancement. Informal listening tests along with objective measures such as Itakura-Saito and log-area-ratios show improvement over the unconstrained technique. By using the Line Spectral Pair transformation, a modest increase in computational requirements results in significant improvement in speech quality. This approach to pole movement constraints is quite robust over direct methods applied to pole radial/angular movements. Finally, this approach may be useful in enhancement for human listeners as well as a preprocessor for speech recognition.

References

- [1] J.S. Lim, A.V. Oppenheim, "All-Pole Modeling of Degraded Speech," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-26, pp. 197-210, June 1978.
- [2] J.H. Hansen, M.A. Clements, "Enhancement of Speech Degraded By Non-White Additive Noise," Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [3] J.R. Crosmer, "Very Low Bit Rate Speech Coding Using the Line Spectrum Pair Transformation of the LPC Coefficients," Ph.D. dissertation, School of Electrical Engineering, Georgia Institute of Technology, Atlanta, June 1985.
- [4] S.R. Quackenbush, "Objective Measures of Speech Quality," Ph.D. dissertation, School of Electrical Engineering, Georgia Institute of Technology, Atlanta, May 1985.
- [5] S.F. Boll, "Suppression of Acoustic Noise in Speech using Spectral Subtraction," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-27, pp. 113-120, April 1978.

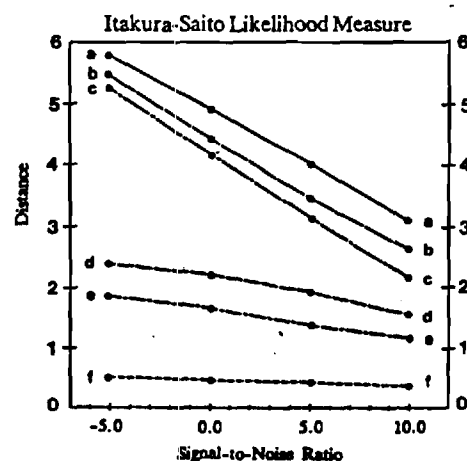


Figure 4: Comparison of enhancement algorithms over SNR.

- a.) Original Distorted Speech
- b.) Boll: Spectral Subtraction, using magnitude averaging
- c.) Lim-Oppenheim: Unconstrained Wiener filtering
- d.) Hansen-Clements: employing Inter-Frame constraints
- e.) Hansen-Clements: employing Inter & Intra-Frame constraints
- f.) Theoretical limit: using undistorted LPC coefficients, *a*.

HIDDEN MARKOV MODEL SPEECH RECOGNITION BASED ON KALMAN FILTERING

Mark A. Clements and Sungjae Lim

School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Abstract

Traditional hidden Markov model speech recognition is generally based on a set of parameters (often LPC related) which are extracted at discrete intervals. Such an analysis necessitates use of a discrete-trial hidden Markov model in which the underlying states can only change at intervals related to the frame rate of the analysis. The exact locations of the analysis windows used can influence the front-end outputs and as a result can cause confusion between words differing in short-duration consonants. In the current study, an alternate method which does not require segmentation is proposed, and a simple version is implemented. The discrete trial hidden Markov model algorithms are adapted to this framework leading to significantly improved recognition performance.

Introduction

Over the past few years, the method of choice for many speech recognition applications has been based on hidden Markov modeling. Steady improvement has been reported in such areas as peakier independence, noise handling, training and response times, as well as general performance. The first HMM based systems modeled speech as a discrete state discrete trial Markov process with discrete observations. Recently, new models which allow a continuous distribution of observations have been presented. Although the methods for accomplishing this differ, they all eliminate the vector quantization step and virtually all report improved performance. Throughout all these models, however, the assumption remains that sampling the parameterization of the speech (e.g., spectral or LPC based parameters) is only necessary every 10 to 30 milliseconds. When words differ only by a short duration interior consonant, however, the exact placement of the analysis windows can have an impact on performance. The current study is the result of an attempt to eliminate these windowing effects in an efficient manner. The front-end is based on a Kalman filtering model which produces an output for every sample point. The matching algorithm can be considered an approximation to a discrete state continuous transition hidden Markov modeling technique.

Front-End Analysis

A general autoregressive representation of speech can be used on a model of the form:

$$\mathbf{x}(k) = \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{B}(k)\mathbf{u}(k) + \mathbf{G}(k)\mathbf{w}(k) \quad (1a)$$

$$\mathbf{y}(k) = \mathbf{C}^T(k)\mathbf{x}(k) + \mathbf{v}(k) \quad (1b)$$

where the vector $\mathbf{x}(k) = [x(k), x(k-1), \dots, x(k-p+1)]^T$, $x(k)$ is speech without noise, $\mathbf{u}(k)$ is a periodic input and $\mathbf{B}(k)$ gain, $\mathbf{w}(k)$ a noise input and $\mathbf{G}(k)$ its gain, $\mathbf{C}(k)=[1, 0, 0, \dots]$ representing a vocal-tract filter. The sequence $\mathbf{y}(k)$ is the digitized speech and is the same as $\mathbf{x}(k)$ with no observation noise.

Systems similar to this have been used to model many varied signals arising in innumerable applications. In the linear prediction synthesis model $\mathbf{A}(k)$ remains constant over 10 to 30 millisecond intervals, one of $\mathbf{u}(k)$ or $\mathbf{w}(k)$ is usually set to zero, and $\mathbf{v}(k)$ is zero. In the LPC analysis model, $\mathbf{u}(k)$ and $\mathbf{v}(k)$ are generally assumed to be zero so that $\mathbf{A}(k)$ can be estimated every 10 to 30 milliseconds. Recursive linear least squares estimation based on the general model falls within the general area of Kalman filtering, which allows one to efficiently compute the least squares estimate of $\mathbf{x}(k)$ from the least squares estimate of $\mathbf{x}(k-1)$ and $\mathbf{y}(k)$. The property we wish to exploit is that if we have modeled the system correctly, the prediction error, $\mathbf{v}(k)$, would be white. Even if the system model is correct, the prediction error signal $\mathbf{v}(k)$ will not be zero due to the noise terms. It should have a predictable ratio of its power to the unfiltered signal's power, however. If there are L possible models from which the observed signal was generated, this idea can be used for computing the relative likelihoods of each model given the observed signal. Denote $\mathbf{v}_i(k)$ the prediction residual (innovations sequence) for system i given observations $\mathbf{y}(1), \mathbf{y}(2), \dots, \mathbf{y}(k)$, and $p_i(k)$ the probability of system i given $\mathbf{y}(1), \mathbf{y}(2), \dots, \mathbf{y}(k)$. It has been shown [1] that

$$p_i(k+1) = \frac{N(\mathbf{v}_i(k), \mathbf{V}_i(k))p_i(k)}{\sum_j N(\mathbf{v}_j(k), \mathbf{V}_j(k))p_j(k)} \quad (2)$$

where $\mathbf{V}_i(k)$ is the variance of $\mathbf{v}_i(k)$ if model i is correct, and $N(a,b)$ represents the Gaussian density of mean zero and variance b evaluated at a . Computation of $\mathbf{V}_i(k)$ and $\mathbf{v}_i(k)$ can be performed recursively using the Kalman filtering equations, with $\mathbf{V}_i(k)$ computed off-line. It should be noted that if $\mathbf{v}(k)$ and $\mathbf{u}(k)$ are set to zero, and $\mathbf{A}(k)$ and $\mathbf{G}(k)$ are allowed to change only at abrupt intervals, then $\mathbf{v}_i(k)$ becomes an LPC residual for model i .

Choosing the value of i which maximizes $p_i(k)$ is in many ways like implementing an LPC vector quantizer. Several important differences exist, however. First, different forms of the models can be used. This would, for example, allow different order models for different sounds. Second, periodic components could specifically be put into some models through $\mathbf{u}(k)$. Third, additive colored noise can be modeled explicitly. Fourth, $\mathbf{A}(k)$ can be a time varying transition matrix which could be used in a manner similar to matrix quantization in speech coding. And fifth, the probability calculations via equation (2) is not merely an Itakura distance. The above described procedure has been applied with success to cardiac arrhythmia detection [2] and stochastic aircraft control [3].

Despite the potential power of this technique, a number of difficulties remain. Most notable is the training procedure for the models appropriate to speech. Since this research was intended mainly to study the effects of eliminating the windows

from the speech analysis, a greatly simplified model was adopted. The adoption of this simplified form of equations (1) should in no way be construed to mean that more elaborate models could not be trained, but merely that it was deemed more appropriate to use simple models in a first set of experiments. The models used were:

$$\mathbf{x}(k) = \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{G}\mathbf{w}(k) \quad (3a)$$

$$y(k) = \mathbf{C}^T \mathbf{x}(k) \quad (3b)$$

where $\mathbf{w}^T(k) = [w(k), w(k-1), \dots, w(0)]$ and the sequence $w(k)$ is white Gaussian noise of zero mean and variance q which is uncorrelated with all values of $x(m)$, $m < k$.

$$\mathbf{A} = \begin{bmatrix} a(1) & a(2) & \dots & a(p) \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} \quad (4)$$

$$\mathbf{C}^T = [1, 0, 0, \dots] \quad (5)$$

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix} \quad (6)$$

$$\mathbf{Q} = \begin{bmatrix} q & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix} \quad (7)$$

The Kalman filtering equations become

$$\begin{cases} \nu(k) = y(k) - \mathbf{C}^T \hat{\mathbf{x}}(k|k-1) \end{cases} \quad (8a)$$

$$\begin{cases} \hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{M}(k)\nu(k) \end{cases} \quad (8b)$$

$$\begin{cases} \hat{\mathbf{x}}(k+1|k) = \mathbf{A} \hat{\mathbf{x}}(k|k) \end{cases} \quad (8c)$$

$$\begin{cases} \mathbf{V}(k) = \mathbf{C}^T \mathbf{P}(k|k-1) \mathbf{C} \end{cases} \quad (8d)$$

$$\begin{cases} \mathbf{M}(k) = \mathbf{P}(k|k-1) \mathbf{C}^T \mathbf{V}^{-1}(k) \end{cases} \quad (8e)$$

$$\begin{cases} \mathbf{P}(k|k) = \mathbf{P}(k|k-1) - \mathbf{M}(k) \mathbf{C}^T \mathbf{P}^T(k|k-1) \end{cases} \quad (8f)$$

$$\begin{cases} \mathbf{P}(k+1|k) = \mathbf{A} \mathbf{P}(k|k) \mathbf{A}^T + \mathbf{G} \mathbf{Q} \mathbf{G}^T \end{cases} \quad (8g)$$

where $\hat{\mathbf{x}}(i|j)$ is the best estimate of $\mathbf{x}(i)$ given $y(0), y(1), \dots, y(j)$; and $\mathbf{P}(i|j)$ is the covariance matrix of $\mathbf{x}(i)$ given $y(0), y(1), \dots, y(j)$. If white observation noise of variance Z were included, equation (8d) would merely have a Z added. The last four of these equations are computable off-line. The only training that needs to be done is for the \mathbf{A} matrices, which are all of the same form. It was decided that the values $a(1), a(2), \dots, a(p)$ for all L models could be computed by training a p th order LPC vector quantizer with L codewords. In this study, L was 64 and p was 10. The vector quantizer training was done on speech produced by one talker speaking roughly 40 seconds of continuous speech. Twenty millisecond Hamming windows were applied every 10 milliseconds. Vector quantizer training was accomplished using a binary split algorithm. It may be seen at this point that the use of windows and frames in training is

counter to the goals of the model. However, since so many frames were used in training (40,000), and since such a large number of different frame positions were sampled, a highly representative set of systems was undoubtedly compiled.

The recursion of equations (8) were initialized using

$$\begin{bmatrix} R_0 & R_1 & R_2 & \dots & R_9 \\ R_1 & R_0 & R_1 & \dots & R_8 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R_9 & R_8 & R_7 & \dots & R_0 \end{bmatrix} \quad (9)$$

where R_i is the i th autocorrelation lag.

The variance in equation (2) is linear with respect to the signal power, and should be normalized. This was done using a second order window on the squared signal whose z -transform was

$$\frac{1}{(1-0.961z^{-1})^2} \quad (10)$$

The time constant was roughly 10 milliseconds.

Another issue dealt with how the system should be constrained so that it would not "lock on" to a specific model and make it difficult to track a change of model. Following [2], we set limits on the maximum and minimum values $p(k)$ could achieve. A post-analysis of results, however, showed this step to be of little consequence in the current system.

The system described above was implemented and used to analyze some continuously spoken sentences. The output of the analysis consisted of one 6-bit number for every speech sample analyzed (8,000/second). Several observations are worth pointing out at this time. First, long runs of the same codeword (greater than 10 milliseconds) were usually seen in fairly steady-state portions of the speech. However, in transition and many consonant regions, codeword runs could be very short (less than 2 milliseconds). Second, voiced sounds usually produced a string of codewords which were one value for most of the pitch period, and another for the remainder of the period. This behavior seems reasonable since periodic input was not put into any of the models. Third, speech could be synthesized from the strings of codewords by simply inputting noise into equations (3) and allowing the filter to change every sample. Although poor in quality, this speech was superior to that produced by a conventional 6-bit LPC vector quantizer updated every 10 milliseconds.

Hidden Markov Modeling

In some versions of the hidden Markov model speech recognizers, a vector quantizer codeword every 10 to 30 milliseconds is all the recognizer or training procedure sees. The system of the current study can supply a recognizer with a codeword every speech sample and hopefully eliminate windowing artifacts. Unless a method could be formulated which would reduce the computational burden by at least an order of magnitude, however, such an expansion of data would be quite unmanageable. Such a method which exploits the fact that codewords often come in long bursts will be presented below.

Notation

Due to the use of certain symbols in the earlier part of this paper, it is necessary to use some non-standard notation to avoid confusion. For a discrete state discrete transition discrete observation hidden Markov model, we must define:

n = number of states

N_v = number of possible observations (vectors) = 64

Π_i = probability the model starts in state i , and

$$\Pi^T = [\Pi_1, \Pi_2, \dots, \Pi_n]$$

T = transition probability matrix, where :

T_{ij} = probability of transit from state i to state j in one trial; $i=1, \dots, n$; $j=1, \dots, n$.

B = observation probability matrix where

b_{jk} = probability of observing codeword k given state $j = b_j(k)$.

$O(t)$ = codeword observed at time t .

$R(t)$ = observation matrix, consisting of:

$$R(t) = \text{diag}[b_1(O(t)), b_2(O(t)), \dots, b_n(O(t))]$$

The transitions were constrained to be left-to-right making T upper triangular.

For a given model M , and observations $O(1), O(2), \dots, O(F)$, we define

$$\alpha^T(t) = [\alpha_1(t), \dots, \alpha_n(t)]$$

$\alpha_i(t)$ = $\text{prob}[O(1), O(2), \dots, O(t) \text{ / state } i \text{ at } t]$

$$\beta^T(t) = [\beta_1(t), \dots, \beta_n(t)]$$

$\beta_i(t)$ = $\text{prob}[O(t+1), O(t+2), \dots, O(F) \text{ / state } i \text{ at } t]$

$$\Pr[O(1), O(2), \dots, O(F)] = \sum_{i=1}^n \alpha_i(t) \beta_i(t) \quad (11)$$

In matrix form

$$\Pr[O(1), O(2), \dots, O(F)] = \Pi^T R(1) T R(2) \dots T R(F) \beta(F) \quad (12)$$

$$\alpha^T(t) = \Pi^T R(1) T B(2) \dots T R(t) \quad (13)$$

$$\beta^T(t) = T R(t+1) T R(t+2) \dots T R(F) \beta(F) \quad (14)$$

In the left-to-right model

$$\Pi^T = [1, 0, \dots, 0] \text{ and } \beta^T(F) = [0, 0, \dots, 1] \quad (15)$$

Recognition Stage

In the recognition stage, the goal is to find which Markov model is most likely given the sequence of observations: $O = [O(1), O(2), \dots]^T$. The computation is reduced basically to finding $\text{prob}[O/M]$ for each model. For this only the set of matrix multiplies in equation (12) need to be carried out.

Throughout this probability calculation, the matrix T remains constant. Also, the matrices $R(t)$ are the same for many consecutive values of t . Consider now equation (12) decomposed as follows:

$$\Pr[O] = \Pi[R(1) T R(2) \dots R(i) T] \quad (16)$$

$$[R(i+1) T R(i+2) T \dots R(j) T]$$

$$[\dots][\dots][\dots T R(F) \beta(F)]$$

where the sets of matrices within the square brackets correspond to times over which the observations remain constant. The products can be evaluated quite efficiently. Consider the partial product

$$[R(i+1) T R(i+2) T \dots R(j) T] \quad (17)$$

which is equal to

$$[R(j) T]^{j-i} \quad (18)$$

The matrix T is upper triangular and $R(j)$ is diagonal. Therefore $R(j)T$ is upper triangular. If the diagonal elements of $R(j)T$ are distinct, then it can be diagonalized such that:

$R(j)T = P D P^{-1}$ where D is diagonal with its elements the

same as the diagonal elements of $R(j)T$. Therefore:

$$[R(j)T]^{j-i} = P D^{j-i} P^{-1} \quad (19)$$

Stated in terms of the partial forward probabilities:

If $O(t+1) = O(t+2) = \dots = O(t+m)$, then

$$\alpha(t+m) = \alpha(t) [T R(m)]^m = \alpha(t+m) = \alpha(t) P D^m P^{-1} \quad (20)$$

Since the matrix P is comprised of orthonormal eigenvectors, its inverse is merely its transpose. Also, since the eigenvalues are the diagonal entries in the upper triangular matrix, the eigenvectors are obtainable by an efficient recursion. Throughout this procedure, as in other hidden Markov model based systems, scaling must be done to ensure no underflow.

Training Stage

For training, we use a variant of the Baum-Welch reestimation procedure. At each iteration, T_{ij} and $b_j(k)$ are estimated based on the previous estimates and the observations. In summary

$$T_{ij} = \frac{\gamma_{ij}}{\gamma_i} \quad (21)$$

$$b_j(k) = \frac{\sum_{t \in O(t)=k} \alpha_j(t) \beta_j(t)}{\sum_{t=1}^F \alpha_j(t) \beta_j(t)} \quad (22)$$

$$\text{where } \gamma_{ij} = \frac{1}{p} \sum_{t=1}^{F-1} \alpha_j(t) T_{ij} \beta_j((O(t+1))) \beta_j(t+1) \quad (23)$$

$$\text{and } \gamma_i = \sum_{j=1}^n \gamma_{ij} \quad (24)$$

Here γ_{ij} is the expected number of transitions from state i to j , and γ_i is the number of transitions out of state i .

If these equations were used directly, a large computational burden would exist, since the sequence of observations is so long. However, all equations (21) through (24) denote are sample averages. We therefore do not need to compute the sums over all possible terms, but rather over only a sampling. Denote the modified terms by:

$$\gamma_{ij}^M, T_{ij}^M, \text{ and } b_j^M(k).$$

$$\gamma_{ij}^M = \frac{1}{P} \sum_{\tau=1}^{(F-1)/\tau} \alpha_i(\tau) T_{ij} b_j((0(\tau+1)) \beta_j(\tau+1)) \quad (25)$$

$$\gamma_i^M = \sum_{j=1}^F \tilde{\gamma}_{ij} \quad \text{and} \quad (26)$$

$$T_{ij}^M = \frac{\gamma_{ij}^M}{\gamma_j^M} \quad (27)$$

Please note that we are now sampling equation (23) every τ units. Similarly,

$$b_j^M(k) = \frac{\sum_{t \in (0(\tau+1))=k} \alpha_j(\tau) \beta_j(\tau)}{\sum_{t=1}^{F/\tau} \alpha_j(\tau) \beta_j(\tau)} \quad (28)$$

In equation (20) we showed how to compute the values of $\alpha(t+m)$ from $\alpha(t)$ if $0(t+1) = 0(t+2) = \dots = 0(t+M)$.

In a similar manner,

$$\beta(t) = [P_{t+1} D_{t+1}^m P_{t+1}^{-1}] \beta(t+m) \quad (29)$$

if $0(t+1) = 0(t+2) = \dots = 0(t+m)$.

Therefore, the recursions for computing the forward and backward partial probabilities can be performed efficiently.

Experiments

To evaluate the system, we used a set of isolated nonsense words differing only in an interior consonant. We chose this task due to the difficulty often encountered in identifying such utterances, and also because we felt our overall procedure was formulated to solve just such problems. The set of words were of the form /i/-C-/i/ where C represents a consonant, including: /b,d,g,p,t,k,r,w,l,j,s,f,\theta,z,v,\gamma,ts,dz,h,m,n/. These 23 words were spoken twenty times each in two separate sessions (one for training, one for testing) by one male talker. The utterances were filtered and digitized with 12-bit precision at an 8KHz sampling rate.

In parallel with the development system to be tested, a more standard HMM recognition system which accepted as input 6-bit vector quantizer codewords every 10 milliseconds was trained and tested. (This latter system was a highly debugged piece of software developed for other purposes over the last two years.) In both systems, five state left-to-right models were used.

In the standard system, 46 errors were recorded for 90% correct. Twenty-seven of these errors were fricatives being confused with other fricatives. In the new system, 7 errors were recorded for 98.5% correct over the same data. The errors were too few to see a clear trend.

Discussion

We have explored one possible method for approximating a continuous transition hidden Markov model for speech recognition. An important component of this method was to allow a virtually continuous stream of input to be input to the recognizer. The Kalman filter approach is but one of many methods which could be used. We freely admit that we have not at this point explored very deeply into the many variations possible in the Kalman filter model, however. Although our various simplifications led to increased efficiency in the recognition algo-

rithm, the computation time is still a few times larger than with the standard procedure.

On so small a data-base as we have been working, conclusions are perhaps hard to draw. The reduction of the error rate by a factor of 6.5, however, strongly suggests the new method is superior. Although no direct tests were run using continuous density observation probabilities, reported improvements over discrete observation probability models are usually by a much smaller factor.

References

- [1] Lainiotis, D. G., and Park, S. K., "On joint detection estimation, and system identification discrete data case," *Int. Jour. Control*, vol. 17, no. 3, 1973, pp. 609-633.
- [2] Gustafson, D. E., Willsky, A. S., and Wang, J. Y., "ECG/VCG rhythm diagnosis using statistical signal analysis: I. identification of persistent rhythms," *IEEE Trans. Biomed. Eng.*, vol. BME-25, no. 4, July 1978, pp. 344-353.
- [3] Athans, M., Dunn, K. P., Greene, S. C., Lee, W. H., Sandell, N. R., Segall, II, and Willsky, A. S., "The stochastic control of the F-8C aircraft using the multiple model adaptive control (MMAC) method," *Proc. IEEE Decision and Control Conf.*, 10-12, Dec. 1975.

This work was sponsored in part by DoD.

DECEMBER 1986

Contract MDA 904-85-K-0005

	<u>Budget</u>	<u>Expended</u>	<u>Balance</u>
Personal Services	27,692	27,277	415
Fringe Benefits	1,846	2,260	(414)
Materials and Supplies	1,000	930	70
Travel	2,000	1,270	730
Total Direct	32,538	33,265	(727)
Overhead	20,661	20,207	454
TOTAL	53,199	52,672	527

Research in Digital Speech Processing

Progress Report

April - June 1987

Three major components of the project to date have been:

1. Investigation of improved enhancement techniques,
2. Recognition of components of speech, and
3. application of Hidden Markov Modeling.

1 Enhancement

Please find attached a comprehensive summary of our work to date on speech enhancement. (It is copied from a portion of J. Hansen's thesis proposal). In summary, we have been able to demonstrate a method which by all measures we have employed, yields superior results to other single microphone methods. In an earlier report, we stressed that the new technique, which employs spectral constraints, has better performance than spectral subtraction or the Oppenheim-Lim unconstrained methods. We have recently demonstrated that the optimum terminating iteration

is consistent across the new method, over a wide range of SNR's and speech characteristics (see pages 104 and 105), making it a much more viable technique. In addition, we have demonstrated its effectiveness in a colored noise environment.

2 Recognition of Components of Speech

The continuous transition hidden Markov model has been expanded in several ways. First, front-ends other than Kalman filtering have been implemented. Recursive least squares, recursive autocorrelation function estimation, and inverse filtering have all been used. Kalman filtering still appears to be the best method, especially in the presence of additive noise. A new HMM training method which is more efficient has been formulated and is being implemented.

3 Application of Hidden Markov Modeling

We have been successful in synthesizing intelligible speech using 1.8 bits per spectral frame, or 180 bits/second, using a global HMM. More efficient computational methods are being implemented currently.

JUNE 1987

Contract MDA 904-85-K-0005

	Budget	Expended	Balance
Personal Services	51,010	30,230	20,780
Fringe Benefits	3,519	2,956	562
Materials and Supplies	2,500	1,055	1,445
Travel	3,500	1,498	2,002
Total Direct	60,529	25,740	24,789
Overhead	38,435	22,695	15,740
TOTAL	98,964	58,435	40,529

Georgia Institute of Technology

College of Engineering
School of Electrical Engineering
Digital Signal Processing Laboratory
Atlanta, Georgia 30332



GEORGIA TECH 1885-1985

DESIGNING TOMORROW TODAY

March 15, 1988

Mr. Dave Kemp
NSA
Mail Stop R556
Ft. Meade, Maryland 20755

Dear Mr. Kemp:

For the progress report for Fall Quarter 1987 for research contract MDA-904-85-K-0005, we are enclosing a copy of Eric Farges' thesis which resulted from the support.

Sincerely yours,

(' >
Mark A. Clements
Assistant Professor

MAC:kcg

Enclosure

December 31, 1987

Contract MDA 904-85-K-0005

	Budget	Expended	Balance
Personal Services	51,010	41,622	9,388
Fringe Benefits	3,519	4,312	(794)
Materials and Supplies	2,500	1,074	(1,426)
Travel	3,500	1,688	1,832
Equipment	0	0	0
Total Direct	60,529	48,676	11,853
Overhead	38,435	30,457	7,978
TOTAL	98,964	79,133	19,831

Georgia Institute of Technology
College of Engineering
School of Electrical Engineering
Digital Signal Processing Laboratory
Atlanta, Georgia 30332



January 15, 1989

Mr. Dave Kemp
NSA
Mail Stop R556
Ft. Meade, Maryland 20755

Dear Mr. Kemp:

Please find enclosed the progress report for Fall Quarter 1988,
for research contract MDA-904-85-K0005.

Sincerely yours,

Mark A. Clements
Associate Professor

MAC:kcg

Enclosure

Research in Digital Speech Processing

Progress Report

October 1, 1988 - January 1, 1989

The bulk of the work was expended in continuing the large hidden Markov Model development. We have succeeded in writing software to generate a continuous distribution global HMM with 64 states and Gaussian observation statistics. We have obtained the TIMIT database and are beginning to test the system on it.

January 1989

E21-670

Contract MDA 904-85-K-0005

	Budget	Expended	Balance
Personal Services	51,010	49,424	1,586
Fringe Benefits	3,519	5,572	(2,051)
Materials and Supplies	4,200	1,704	2,495
Travel	1,800	1,668	132
Total Direct	60,529	58,368	2,160
Overhead	38,435	36,271	2,163
TOTAL	98,964	94,640	4,324

An analysis-synthesis hidden Markov model of speech

A THESIS

Presented to

The Faculty of the Division of Graduate Studies

By

Eric P. Farges

In Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy in Electrical Engineering


Georgia Institute of Technology

November, 1987


Copyright ©1987 by Eric P. Farges

An analysis-synthesis hidden Markov model of speech

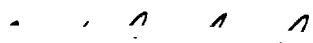
Approved:



Mark A. Clements, Chairman



Abraham H. Haddad



Ronald W. Schafer

Date approved by Chairman 13-NOV-87

A mes parents

ACKNOWLEDGMENTS

I would like to express my gratitude to Dr. Mark Clements for being my Ph.D. thesis advisor and providing the financial support and guidance necessary to carry out the research described here. I would also like to thank Dr. Thomas Barnwell for initially providing the funding for my graduate research assistantship.

I also express my appreciation to Dr. Thomas Barnwell, Dr. Aubrey Bush, Dr. Atif Debs, Dr. Mark Clements, Dr. John Elton, Dr. Abraham Haddad, Dr. Monson Hayes, and Dr. Ronald Schafer for being at one time or another members of my reading committee, making helpful comments, and formulating stimulating questions. I certainly wish to thank all the faculty members of the DSP group for providing such a good research environment and terrific lab-facilities.

My appreciation goes to Dave Dranchak for being so kind to record the long speech database used throughout this research. Thanks to Dr. Joel Crosmer for letting me use his vector quantization software. My special thanks to the following graduate students who were willing to subject themselves to the subjective speech quality test: Bryan George, François Malassenet, David Pepper, and Richard Rose.

I also thank Charles Gimarc and Tom Marsh for helping with the use of the computer facilities.

Finally I wish to express my appreciation to Dr. Hans Puttgen, Dr. Dale Ray, Dr. Thomas Barnwell, and Dr. Ronald Schafer for introducing me with the Georgia Institute of Technology and helping me when it came to dealing with the Georgia Tech administration concerning the settlements for my French military service as a "coopérant scientifique."

Contents

List of Tables	vi
List of Figures	vii
Summary	x
1 Introduction	1
2 Description of hidden Markov models	15
2.1 Stochastic modelling	15
2.2 Basic description of a discrete HMM	17
2.2.1 The model	17
2.2.2 Graphical representations	19
2.2.3 The steady-state distribution of the states	19
2.3 The adjoined HMM	22
2.3.1 The model parameters	22
2.3.2 Likelihood and distance measure	26
2.4 Entropy, bit rate, and structure	27
2.4.1 The concept of entropy	27
2.4.2 Entropies of an HMM	29
2.5 Fundamental issues of HMM's	31
2.6 Other types of HMM's	34
2.6.1 Continuous hidden Markov models	34

2.6.2	Hidden semi-Markov models	35
3	Maximum likelihood estimation of the parameters of hidden Markov models	37
3.1	Presentation of the problem	37
3.2	The theory of the forward-backward algorithm	41
3.2.1	General results	41
3.2.2	Historical development of the algorithm	42
3.2.3	Formulation and interpretation of the growth transformation	44
3.2.4	Recursive evaluation of the growth transformation	48
3.2.5	Computational complexity of the basic FBA	50
3.3	Practical implementation of the FBA	56
3.3.1	Introduction	56
3.3.2	Scaling the forward and backward probabilities	57
3.3.3	Reestimation in terms of the scaled variables	62
3.4	Estimation of other types of HMM's	76
3.4.1	Estimation of the parameters of CHMM's	76
3.4.2	Estimation of the parameters of HSMM's	77
3.5	Evaluation of the forward-backward algorithm	77
3.5.1	Evaluation of the correctness of the algorithm	77
3.5.2	Evaluation of the modelling capabilities of the algorithm . .	80
4	Maximum likelihood trellis decoding	83
4.1	Presentation of the problem	83
4.2	The trellis decoding algorithm	86
4.2.1	The algorithm	86
4.2.2	The issue of the initial node	92
4.2.3	Algorithm complexity	97
4.3	The concept of convergence nodes	101
4.4	Sub-optimum trellis decoding algorithms	110

4.4.1	Backward pruning	111
4.4.2	Forward pruning	114
4.5	Constrained trellis decoding algorithm	118
4.6	State and observation decoding for speech	119
4.6.1	State decoding	119
4.6.2	Observation decoding	120
4.6.3	Estimation of the inverse HMM λ^{-1}	129
5	HMM's and speech coding	133
5.1	The Markov-VQ	133
5.2	The partitioned HMM-VQ	134
5.3	The HMM-VQ	134
6	Experimental results	137
6.1	Training	138
6.2	State decoding	153
6.3	Speech coder bit rate	158
6.4	Speech intelligibility and observation decoding	161
6.5	State interpretation	171
7	Conclusion	176
	Bibliography	180
	Vita	192

List of Tables

3.1	Number of operations for the basic FBA.	53
3.2	Some peak speeds of commercial computers.	54
3.3	CPU time per iteration of the basic FBA.	55
4.1	Computational cost of the state-decoding trellis algorithm.	99
4.2	Memory cost of the state-decoding trellis algorithm.	100
4.3	Analysis-Synthesis HMM's of Speech: summary.	132
6.1	The FBA: a long optimization process.	140
6.2	Log-likelihood and entropies as a function of the iteration number.	141
6.3	Markov-VQ bit rate.	159
6.4	Bit rate as a function of the pegging period K ($H_A = 1.68$).	160
6.5	Bit rate as a function of the pegging period K ($H_A = 1.84$).	160
6.6	Entropies of the HMM-VQ coder from two different methods of derivation.	160
6.7	LPC log-likelihood distances for the test sentence.	164
6.8	Results of the subjective quality testing.	165

List of Figures

1.1	The message model of human communication.	10
1.2	A block diagram representation of the message model.	11
1.3	Major anatomical structures involved in the production of speech.	12
1.4	Cross section of the vocal tract.	12
1.5	Speech processing unit.	13
1.6	The acoustic processor.	13
1.7	The acoustic pre-processor.	13
1.8	Hierarchical structure of the Linguistic Decoder.	14
1.9	Vocal tract shapes for given English vowels.	14
2.1	State diagram of a 3-state Markov Chain.	20
2.2	Evolution of an HMM system over time.	20
2.3	An HMM trellis.	21
2.4	State entropy of a loosely structured 4-state Markov chain.	32
2.5	State entropy of a highly structured 4-state Markov chain.	32
3.1	Basic FBA (without any scaling).	51
3.2	FBA with partial scaling.	65
3.3	Optimized FBA with partial scaling.	67
3.4	Further-optimized FBA with partial scaling.	68
3.5	FBA with full scaling.	75
4.1	Example of a trellis.	88
4.2	The shortest path of depth 5.	91

4.3	The shortest path is the extension of a survivor.	91
4.4	Selecting the best path extending to X_L	91
4.5	Summary of the TDA procedure (numbers are branch lengths). . .	92
4.6	The basic ML trellis decoding algorithm.	93
4.7	Optimum, unconstrained ML trellis decoding algorithm.	96
4.8	Emergence of the convergence node $\{e\}$	105
4.9	The solution path $\{A,a,b,c,B\}$	105
4.10	A convergence node of weight 2, the node $\{C\}$	105
4.11	Unconstrained ML trellis decoding algorithm with backward pruning.	113
4.12	Unconstrained ML trellis decoding algorithm with forward and natural backward pruning.	116
4.13	Operation of the constrained TDA.	120
4.14	Global operation of the state and observation decodings.	125
4.15	Principle of the estimation of the inverse HMM λ^{-1}	125
4.16	Behavior of the ML observation decoding based on $\tilde{\lambda}$ and λ^{-1}	126
4.17	Global feedback estimation of λ^{-1} , and possibly λ	127
5.1	Range of the coder bit rate as a function of the pegging period. . .	136
6.1	Evolution of the log-likelihood.	145
6.2	Evolution of the entropies.	146
6.3	Evolution of the initial distribution of the states.	147
6.4	Evolution of the steady-state distribution of the states.	148
6.5	Histogram of the entries of the transition matrix A (λ_{100}).	149
6.6	Histogram of the entries of the output probability matrix B (λ_{100}). .	150
6.7	Transition matrix of model 1.	151
6.8	Transition matrix of model 20.	151
6.9	Transition matrix of model 40.	152
6.10	Transition matrix of model 100.	152

6.11 The structure of the transition matrix A	153
6.12 Distributions of the convergence node weight, convergence length, and decoder delay.	155
6.13 Most probable log-spectrum in each of the 64 states.	156
6.14 Expected log-spectrum in each of the 64 states.	157
6.15 Speech waveform of the test utterance.	166
6.16 More detailed view of the speech waveform.	167
6.17 Frame by frame view of the speech waveform.	168
6.18 Spectra of the original and reconstructed speech.	169
6.19 Examples of reconstructed waveforms.	170
6.20 Some typical state structures.	173
6.21 Formant clustering of the states 2, 52, and 58.	174
6.22 State duration distributions.	175

Summary

A computerized speech processing system which would perform in a manner comparable to that of a human being would need to analyze the speech waveform not only as a signal but also as a highly complex encoding of structured knowledge. The great fuzziness of the acoustic waveform hides a deep structure hierarchically organized around linguistic constraints.

This research project attempts to uncover the hidden speech structure from the observed speech waveform automatically, by defining acoustic units and identifying the rules governing the associations and interactions of these units. Due to the complexity of the problem, the very high number of "parameters" to identify and derive, and a linguistic science still in its infancy, a probabilistic and statistical approach was preferred over a deterministic approach. The speech waveform (known as the set of *observations*) and the acoustic units (known as the *states*) go through a double chaining process modelled by a first-order hidden stationary Markov chain. A stochastic model known as a hidden Markov model (HMM) is derived and applied to continuous speech. The following four fundamental problems of HMM's are addressed: estimation of the model parameters, determination of the state structure from the speech waveform, synthesis of intelligible speech from a state structure, and interpretation of the state representation.

Many applications of such a global 64-state, 1024-observation HMM of speech are envisioned: automatic speech segmentation, speech analysis, noisy speech enhancement, and continuous speech recognition. However, only the application of this research to very-low-bit-rate speech coding is discussed in detail. The maximum likelihood estimate of the model is experimentally shown to be also a minimum entropy estimate. Intelligible speech was synthesized from the encoding of the spectral information requiring as little as 1.7 bits/frame. Specific states and state-structures were identified as a consistent description of specific sounds (like vowels) and modelled the actual duration of the sounds quite well.

The code of speech resides in its sounds. Thus a deep understanding of speech communication depends on knowing the sound patterns of the speech code. J.M. Pickett

Ainsi le langage est une simplification complexifiante qui permet d'utiliser une partie de l'hyper-complexité cérébrale, de construire/reconstruire une nouvelle complexité discursive, et ainsi de dialoguer avec la complexité du réel. E. Morin

The key problem of scientific development is the problem of how the structure of human consciousness and language evolve ... V. Nalimov

Ce qui est bien connu, justement parce que bien connu, n'est pas connu. Hegel

Quel beau sujet de dispute sophistique tu nous apportes là, Ménon; c'est la théorie selon laquelle on ne peut chercher ni ce qu'on connaît, ni ce qu'on ne connaît pas: ce qu'on connaît parce que, le connaissant, on n'a pas besoin de le chercher, ce qu'on ne connaît pas parce qu'on ne sait même pas ce qu'on doit chercher. Platon

CHAPTER 1

Introduction

The idea of man-machine communication by speech has been on the minds of scientists and engineers for quite some time [2]. The naturalness and ease of such a communication means makes it attractive. Unfortunately the elusive simplicity of natural speech escapes us when it comes to designing artificial speech processing systems. As stated by Akmajian et al. [4, p.493]:

“Speaking and understanding our native language is so spontaneous and apparently easy that we are completely unaware of the remarkably complicated tasks carried out by the human brain to make it possible for us to use language so freely and effortlessly.”

This remark points out several important facts about speech. Speech is a language, that is, a complex mechanism for encoding information. This encoding mechanism allows reduction, storage, and retrieval of information so that the brain can be provided with an internal representation of concepts describing part of the world human beings deal with. The communicative side of speech can be represented by the *message model of human communication* depicted in figure 1.1. A block diagram representation is given in figure 1.2. The message model of human communication between a speaker (transmitter) and a listener (receiver) can help us identify basic properties of spoken languages. The brain possesses an abstract internal representation of the world. It has the ability to generate and use knowledge and concepts, being possibly triggered by the influence of external stimuli

provided by the human senses. On the speaker side, the process of information coding occurs in the brain: the internal representations are converted in highly complex ways into neural electrical waves sent through the whole nervous system. The neural signals in turn continuously drive the speech production apparatus described in figures 1.3 and 1.4. As a result an acoustic sound wave is produced and transmitted, possibly with corrupting noise, to the listener through the communication channel (for example, open air or a telephone line). The acoustic wave, in terms of signal processing, is represented by the speech waveform. The waveform is a noisy measure, as a function of time, of the acoustic pressure of the air at a given position in space; it is recognized as encoding most of the significant speech information. The speech signal is either the input or the output of classical digital speech processing systems. On the listener side, the process of human communication is reversed, converting the acoustic sound wave into internal representations as shown in figure 1.2.b. For the purpose of man-machine communication, either the speaker can be removed and replaced by a "speaking machine," or the listener can be removed and replaced by a "listening-understanding machine." Here the word "machine" stands for currently available digital computer or special purpose DSP hardware.

A DSP implementation of the listener side of the message model of human communication is shown in figures 1.5, 1.6, and 1.7. The incoming speech signal is analyzed by an Acoustic Processor (AP) to generate acoustic patterns (or states) X_n . These patterns are encoded efficiently into acoustic codewords through the Bit Allocator (BA). The codewords are processed through a Linguistic Decoder (LD) to derive classical linguistic representations of speech (such as phonemes, words, etc). Finally a meaning is derived from the structure of the linguistic unit representations, and encoded using internal representations, to build or augment the internal knowledge database. In this research project, our emphasis will be on the design of the acoustic processor and the bit allocator. Some features pertinent

to the linguistic decoder might also be presented. Our approach will be to separate the acoustic processor into two independent systems: the acoustic pre-processor (APP) and the hidden Markov model processor (HMMP) — see figures 1.6 and 1.7. The acoustic pre-processor — also referred to as the *classical acoustic processor* — samples (and possibly quantizes) the continuous speech waveform at the Nyquist rate. The digitized waveform is analyzed by Linear Predictive Coding (LPC) [95,98,116]: the speech waveform is divided into frames¹ and a parametric model — such as a 10th order all-pole linear filter — is derived for each frame. In each frame the speech is assumed quasi-stationary. Independent successive models for consecutive frames are concatenated together to form the overall time-varying representation of speech. A vector quantizer (VQ) [30,96,151] selects from a finite codebook of parametric representations, the parametric representation closest to the actual representation. This classical acoustic processor, although it models speech as a signal quite efficiently, fails to model speech as a language. For one thing, it basically ignores the time dependencies between consecutive frames. In other words, the classical AP takes advantage of the acoustical (short term) structure of the speech waveform, but not its (long term) linguistic structure. Although recently, speech coding schemes, such as matrix quantization [151] and segment vocoders [129,130,40], have tried to take advantage of the frame dependencies, they lack flexibility and generality. In this research we will try to emphasize the idea that the speech waveform is a representation of a language. A language will be seen as a set of *units* related to one another through a set of *rules*, dependencies, and relationships. Part of the speech knowledge will be encoded into (the nature of) the units themselves, but most of the speech information will be encoded into the relationships between these units. As opposed to a local model of speech based on a single frame, the HMMP will build a global model based on the overall continuous speech structure. As will be seen later, the HMMP will encode the speech knowledge into a network of nodes and connections between these

¹Usually 15ms frames of 120 samples, for a sampling rate of 8kHz.

nodes. To summarize, the HMMP defines and extracts acoustical units and the rules of their associations. The APP generates a representation of the acoustical units. Now, what kind of dependencies are we referring to? These are long term dependencies involving several speech frames. On the contrary, short term dependencies refer to dependencies between samples of the speech waveform, in a single frame, such as the ones modelled by LPC. Short term dependencies relate to the signal aspect of speech, long term dependencies to its language aspect. Not only should we consider the dependency of the present speech on its past, but also on its future. This is known as the “planning ahead” phenomenon. At a given instant in time, a speaker has a “picture in mind” of what he plans to say. Therefore a present sound pattern is influenced by past but also future sound patterns. One consequence of such interactions is coarticulation: one sound pattern is modified and adapted to fit the upcoming next sound, or the next sound is started before the previous one is finished.

Speech structures and dependencies also emerge from *linguistic constraints*. Most linguistic theories [4,94,97,113,137] interpret speech as a language hierarchically structured around the following five *sources of knowledge*:

- Phonology: the study of the structure of the sounds of a language based on a finite number of elementary sound patterns: the phonemes.
- Morphology: the study of the structure of words based on a finite number of constitutive parts known as morphemes.
- Syntax: the study of the proper (“grammatical”) structure of sentences.
- Semantics: the study of meaning in a language.
- Pragmatics: the study of a language use as a means of communication.

These various sources of knowledge interact in a complex way, as noticed by Akmajian et al. [4, p.466]:

“At every level — phonological, morphological, syntactic, semantic, and pragmatic — human language is an intricate system of abstract units, structures, and rules, used in an equally intricate system of communication.”

The use of the sources of knowledge, to derive meaning from speech, is neither sequential nor parallel, but hierarchical. Nalimov [106, p.161] stated:

“One of the most important attributes of language symbol systems is the manifestation of hierarchical structure.”

Including part of this linguistic-like structures into the AP would facilitate the implementation of the LD greatly. A hierarchical block diagram of a LD is shown in figure 1.8. In the message model, when the listener decodes the speech signal, he uses not only the incoming speech waveform, but also his internal knowledge of the linguistics of the language. Similarly when the speaker generates a speech signal, he uses his knowledge of linguistics to constrain the output speech waveform. In other words the production of the speech waveform is partly driven by linguistic constraints. The speech waveform is an observed surface representation of an underlying, hidden, and complex linguistic structure. This research will attempt to identify the reflection of this structure on the acoustic waveform with a global hidden Markov model of continuous speech. As summarized by Levinson [87]:

“...a linguistic theory of speech had emerged according to which all spoken language was viewed as a composition of a relatively small number of primary symbols of which measurable acoustic events were correlates and which could be combined according to well-defined sets of rules. ...Models of this type [HMM's] are particularly appropriate for describing the speech signal since the actual sound pressure wave that we measure is merely an encoding of some underlying symbolic process occurring in the unobservable and completely mysterious recesses of the brain. ...Thus the observable process will attempt to

account for the measured physical correlates of the underlying linguistic structure which is to be explained by the hidden process."

Another important characteristic of a written or spoken language is that it can be represented by a finite set of discrete symbols, the so-called fundamental units. A written language can be described in terms of the letters of the alphabet, in terms of morphemes, words, etc. The sounds of a language can be described in terms of phonemes, diphones, etc. However, the correspondence between these discrete phonemes and their continuous counterpart — their acoustical realization into the speech waveform — is difficult to establish. The concepts formed in the brain are inherently "discrete objects." The speech encoding mechanism transduces them into "continuous objects." The neural signals which drive the speech production apparatus, the acoustic speech wave, and the speech waveform are continuous in nature (at the macroscopic level). However, they represent discrete objects. The speech waveform segmentation — such as the detection of word boundaries, — and the automatic extraction of phonemes from the continuous waveform, is a difficult problem. When listening to a foreign language, for which one does not have a priori knowledge, speech appears as a continuous flow of sounds. Discrete acoustic units, at a deeper level of abstraction than phonemes, should be defined if a speech model is to perform the transduction continuous \leftrightarrow discrete characteristic of a language. The classical sampling and frame division of the speech waveform does not provide the desired discretization of the speech into globally meaningful (i.e., related to language and linguistics) acoustic units, but merely provides a digitization meaningful in terms of a signal, which could be something other than speech. Finally these acoustic units should not be expected to be absolute units, like classical physical units; they should capture the variability of the acoustical realization of the underlying corresponding unit.

To summarize, speech is a complex continuous acoustical encoding of the discrete units, rules, and hierarchical structures of language. An efficient model

of the speech waveform should be able to identify discrete acoustic units from the continuous speech signal. Moreover, these units and the rules governing their associations should be meaningful in terms of the linguistic constraints of the language. This research project will investigate for that purpose a probabilistic model known as a *hidden Markov model* (HMM). An HMM sees the speech waveform as the surface realization of an underlying hidden language structure. The observed speech is described by proper representations of the signal called the *observations*. The hidden structure (the acoustic units) is described by the *states* of a Markovian process. The rules of associations of the states are probabilistic and described by the transition probabilities of the Markov chain. The interaction of the hidden structure with the surface structure is described by a *probabilistic function of the Markov chain* characterized by output probabilities. The evolution of the state sequence should account for the non-stationarity of the speech, by allowing different observation distributions in each state. To achieve the characteristics described above, the model should be globally derived from a long training corpus of continuous speech. The level of abstraction of the states will be the level of the speech frame (15ms frames, i.e., 67 states per second; on the basis of 10 phonemes per second, it represents roughly 7 states per phoneme). Discrete HMM's (i.e., HMM's associated with the APP of figure 1.7) will be investigated with 64 states and 1024 observations. A state, outside from its meaning as an acoustic unit of the language, could also be seen as a characteristic "sample" over time of the possible configurations of the human vocal-tract (see figure 1.9). Our modelling approach is probabilistic and top-to-bottom. The probabilistic formulation can account for a large number of a priori unknown parameters of the language. It does not deny the deterministic aspect of speech, but provides selective choices arranged according to their probabilities of occurrence. The approach is also top-to-bottom in the sense that it starts with the speech waveform at the top, and derives the significant elements of the hidden structures at the bottom. Some of the advantages of that type of modelling are: the automatic identification of the acoustic units and rules

of the language, and the automatic extraction of the acoustic units from the original speech. A disadvantage might be seen in the level of abstraction of the acoustic units, in the sense that an interpretation of the units to a higher level is needed to relate them to the classical linguistic units. On the contrary, a deterministic bottom-to-top model would select and define units and rules (like phonemes) a priori. The advantage resides in the fact that no unit interpretation is necessary. However, the following significant disadvantages are encountered: linguistic units and rules are only partially known, there are many of them, and it is not known which ones are the most significant for a speech processing system; the automatic extraction of these units from the original speech is very difficult. Finally our global HMM approach departs from recent local word-based or phoneme-based HMM approaches, respectively used for isolated word recognition and connected word recognition [7,12,37,73,85,118,140]:

- Our modelling is global and requires a large model (64 states, 1024 observations) directly applied to the whole waveform, as opposed to a small model (5 states, and 64 observations, or 10 states, and 200 observations) indirectly applied to parts of the segmented waveform.
- The modelling is totally automatic and requires no segmentation or labelling of the speech waveform.
- We would like to promote HMM's to the full rank of speech models, not only with analysis, but also synthesis capabilities.

Some of the questions and problems addressed in this research will be the following:

- Is it possible to formulate and derive such a large global HMM of continuous speech? This is already a difficult problem as noted by Miller and Chomsky [41, p.36] for n^{th} order Markov models:

“A staggering amount of text would have to be scanned and tabulated to make reliable estimates.”

- Can an automatic practical computerized estimation of the parameters of the model be performed?
- Can the underlying structure be recovered automatically from the original speech?
- Can intelligible speech be synthesized from a deep structure?
- Can the model be applied to very-low-bit-rate speech coding?
- Can the deep state structure be correlated with higher level classical linguistic structures?

Chapter 2 will formulate an HMM mathematically and derive some of its important features. Chapter 3 will provide the solution of the maximum likelihood estimation problem for the parameters of the model. Chapter 4 will show how the state structure can be recovered from the speech, and inversely how speech can be synthesized from a state structure, through maximum likelihood trellis decoding. Chapter 5 will show how the HMM can be applied to the specific problem of very-low-bit-rate speech coding. The results of experiments on actual continuous speech will be reported in chapter 6.

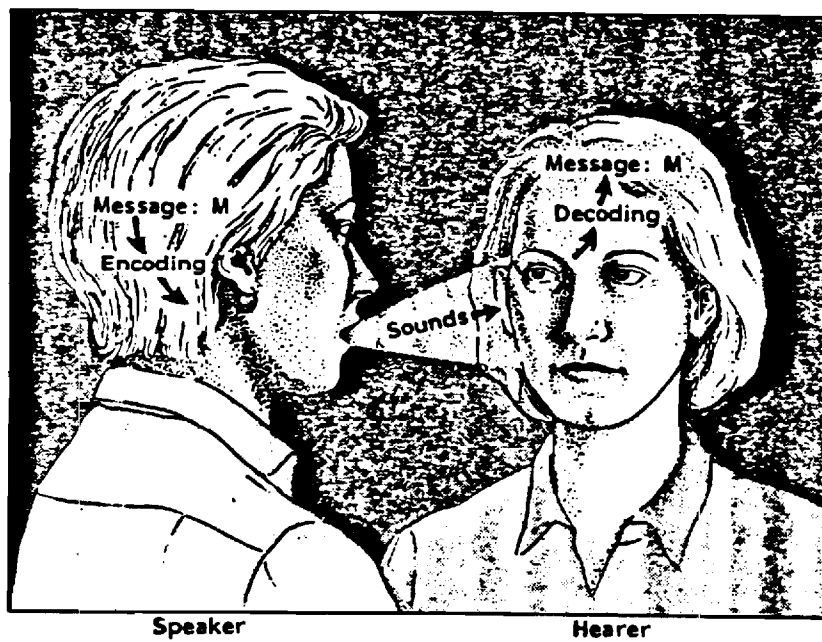
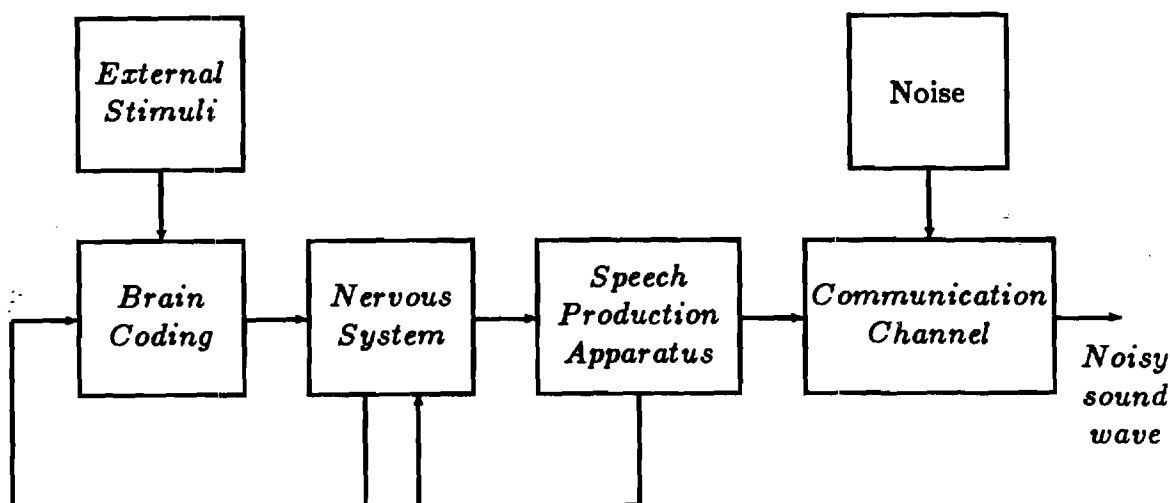
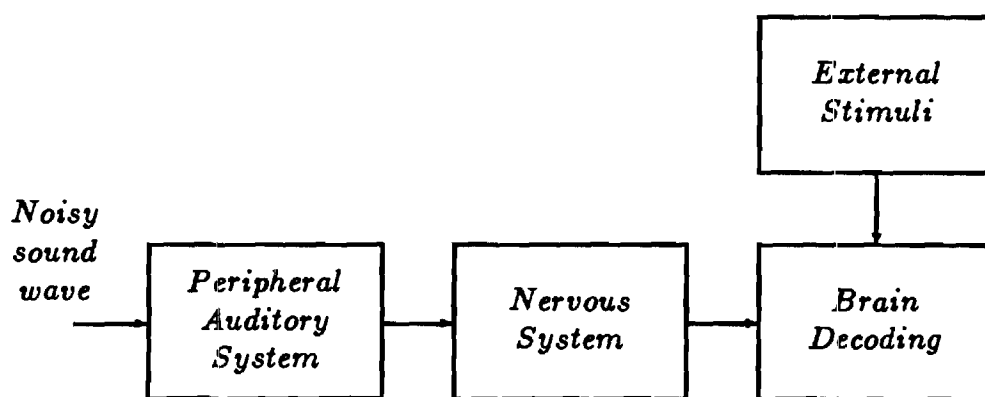


Figure 1.1: The message model of human communication.
(reproduced from [4, p.393])



1.2.a: The speaker (transmitter) side transduces the message internal representations into a sound wave.



1.2.b: The listener (receiver) side transduces a noisy sound wave into internal representations.

Figure 1.2: A block diagram representation of the message model of human communication.

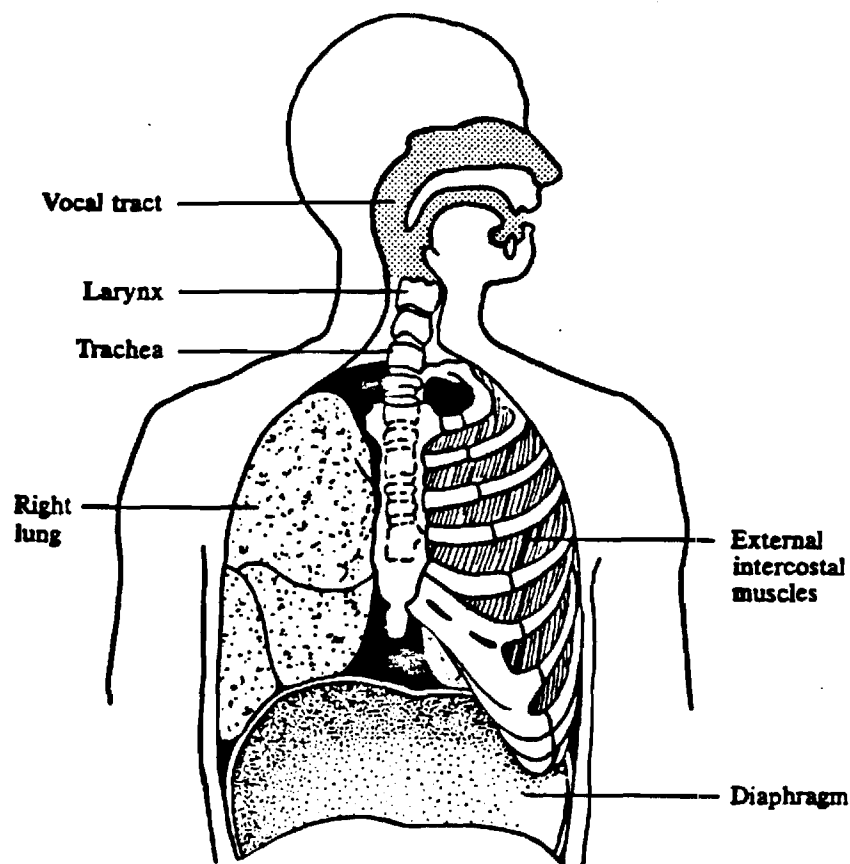


Figure 1.3: Major anatomical structures involved in the production of speech.
(reproduced from [4, p.103])

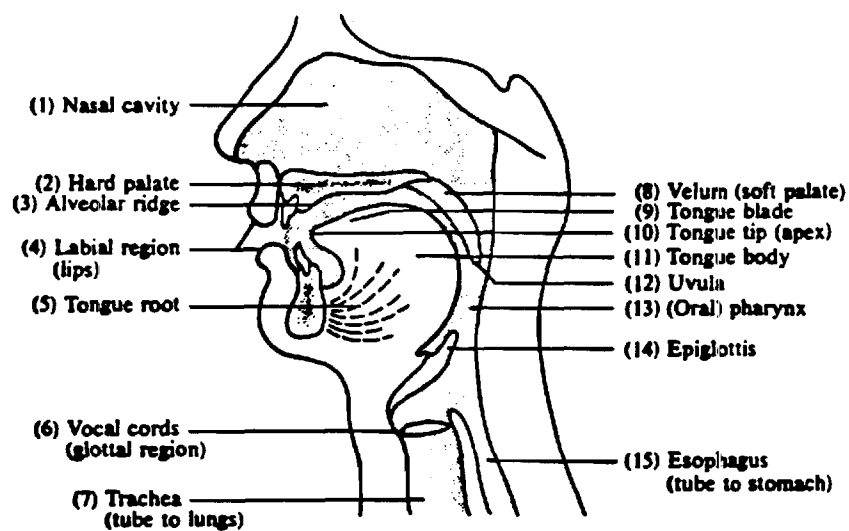


Figure 1.4: Cross section of the vocal tract.
(reproduced from [4, p.109])

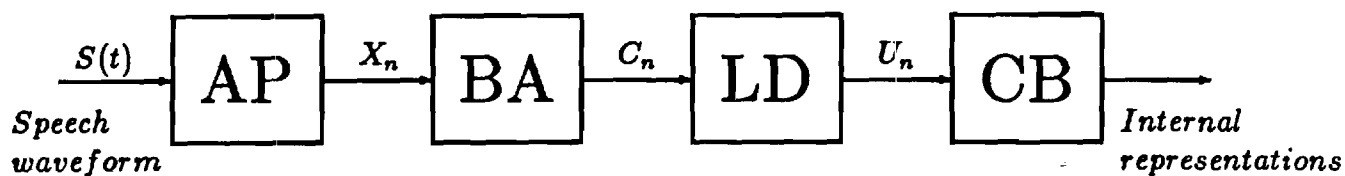


Figure 1.5: Speech processing unit: the DSP implementation of the listener side of the message model of human communication.

AP : Acoustic Processor

BA : Bit Allocator

LD : Linguistic Decoder

CB : Concept Builder

X_n : Acoustic patterns

C_n : Acoustic codewords

U_n : Linguistic units

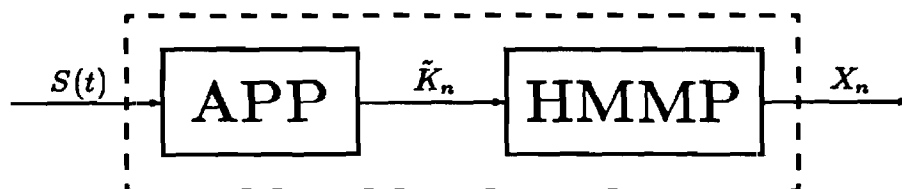


Figure 1.6: The acoustic processor.

APP : Acoustic Pre-Processor

HMMP : Hidden Markov Model Processor

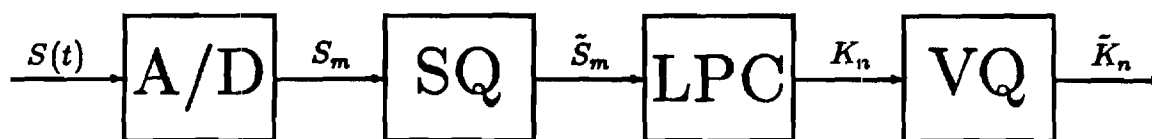


Figure 1.7: The acoustic pre-processor.

A/D : Analog to Digital converter

SQ : Scalar Quantization

LPC : Linear Predictive Coding

VQ : Vector Quantization

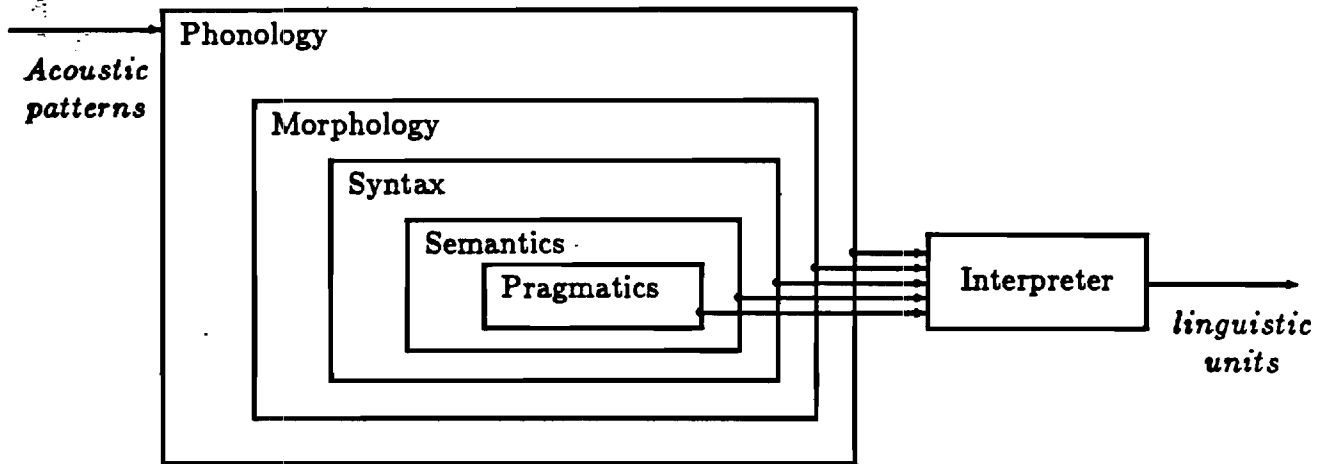


Figure 1.8: Hierarchical structure of the Linguistic Decoder.

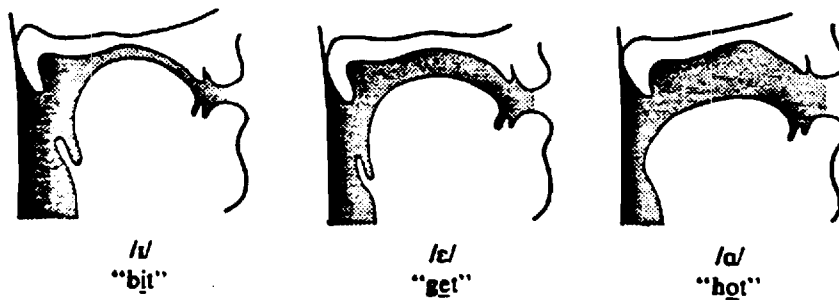


Figure 1.9: Vocal tract shapes for given English vowels.
(reproduced from [4, p.114])

CHAPTER 2

Description of hidden Markov models

2.1 Stochastic modelling

The stochastic modelling [126,24,66] point of view chooses to represent the observed evolution of a non-stationary system by multidimensional random variables Y_t 's. A sequence of the form $\{Y_1, Y_2, \dots, Y_n, \dots, Y_L\}$, denoted by $Y(1:L)$, is a set of samples of the random process $\{Y_t\}$, taken at discrete time intervals. The variable Y_t might take values in a continuous or discrete set. In the case of the acoustical speech waveform, Y_t might be a continuous (non-quantized) parametric representation of the frame t (such as LPC vectors). It might also be discrete representations, such as LPC vectors drawn from a finite codebook. The set of indices, identifying a vector in the codebook, is the Y-alphabet $\mathcal{A}_Y = \{1, 2, \dots, M\}$. The codebook is denoted by $C = \{O_1, O_2, \dots, O_M\}$. The surface behavior of the system is described by $\{Y_t\}$. It is the only information directly available, through observation. The sequence $Y(1:L)$ is therefore called the sequence of *observations*. This observed sequence Y is not, however, the true internal state of the system. Y_t is known to represent a distorted, noisy, fuzzy, uncertain, and encoded version of a more fundamental variable X_t : the underlying state of the system. The *states* X_t 's are also random variables. The possible values of the random variable X_t will be countable and finite: it is the X-alphabet $\mathcal{A}_X = \{1, 2, \dots, s\}$. The production of the observed sequence $Y(1:L)$ is governed (driven) by the non-observable (hidden)

state sequence $X(1:L)$. The true evolution — non-stationarity — of the system is described by the random process $\{X_t\}$. The surface world of the observations is related to the hidden world of the states, through joint and conditional probabilities, $\Pr[X(1:L), Y(1:L)]$ and $\Pr[Y(1:L)/X(1:L)]$ respectively. These probabilities are needed to recover the deep structure $X(1:L)$ from the observed patterns — the analysis phase — and vice versa to generate observations from a given state structure — the synthesis phase. The variables X_t and Y_t are interdependent, and both of them depend on all their past and future values. X_t depends on $X(1:t-1)$ and $X(t+1:L)$; Y_t depends on $Y(1:t-1)$ and $Y(t+1:L)$. However, to be practical, limitations have to be imposed on the stochastic model:

- Stationarity: the stochastic model will be said stationary if all the characteristic probabilities are independent of time.
- Markov assumption: the random process $\{X_t\}$ is Markovian of order δ [1,22,23,26,43,58,60]. The state X_t depends on its past only, the past δ states, i.e.,

$$\Pr[X_t/X(1:t-1), X(t+1:L)] = \Pr[X_t/X(t-\delta:t-1)].$$

- Probabilistic function of a Markov chain: the outcome of the sequence of random variables Y_t 's depends directly and uniquely, in a probabilistic sense, on the states of the Markov chain $\{X_t\}$, i.e.:

$$\Pr[Y_t/Y(1:t-1), Y(t+1:L), X(1:L)] = \Pr[Y_t/X(1:L)].$$

To be practical, especially computationally tractable, this general formulation must be simplified further:

- the Markov chain will be limited to the first order ($\delta = 1$). If one were to use a second order Markov chain, one would need to estimate the s^3 probabilities $\Pr[X_t/X_{t-1}, X_{t-2}]$. In practice, for $s = 64$, roughly 11 hours of speech (on a basis of 15ms frames, and 10 occurrences per estimated parameter) would be needed for

the estimation.

- the production of Y_t will be conditioned on the current state X_t only¹. This type of stochastic modelling is called a *hidden Markov model*. It is primarily based on the following two assumptions:

$$\begin{aligned}\Pr[X_t/X(1:t-1), X(t+1:L)] &= \Pr[X_t/X_{t-1}] \\ \Pr[Y_t/Y(1:t-1), Y(t+1:L), X(1:L)] &= \Pr[Y_t/X_t].\end{aligned}$$

In words, once the state X_{t-1} has been reached, the next state X_t depends only on the present state X_{t-1} , not on the previous states. Similarly, once the state X_t has been reached, the observation Y_t depends only on the current state X_t , not on the previous states, nor on the previous observations. A hidden Markov model (HMM) will allow a very general and powerful description of the speech waveform.

2.2 Basic description of a discrete HMM

2.2.1 The model

An HMM, as introduced above, is described by a stationary probabilistic function of a Markov chain, where the state process $\{X_t\}$ is a first order stationary Markovian process, and the observation process is directly “linked” to the state process. An HMM is uniquely defined by the following three sets of parameters:

- the initial state-distribution column vector: it defines the probability of starting the Markov chain in a given state:

$$\pi_1 = (a_i)_{i=1,s} \quad (2.1)$$

$$\forall i = 1, s \quad a_i = \Pr(X_1 = i) \quad (2.2)$$

$$\sum_{i=1}^s a_i = 1 \quad (2.3)$$

¹For some small models, Y_t can also be conditioned on the transition between two states, i.e., on the states X_{t-1} and X_t .

- the transition probability matrix: it defines the probability of jumping from one state, at any time n , to the next state, at time $n + 1$:

$$A = (a_{ij})_{i=1,s; j=1,s} \quad (2.4)$$

$$\forall n \quad \forall i = 1, s \quad \forall j = 1, s \quad a_{ij} = \Pr(X_{n+1} = j / X_n = i) \quad (2.5)$$

$$\forall i = 1, s \quad \sum_{j=1}^s a_{ij} = 1 \quad (2.6)$$

- the output probability matrix: it defines the probability of producing a given observation, in a given state, at any time n :

$$B = (b_{ik})_{i=1,s; k=1,M} \quad (2.7)$$

$$\forall n \quad \forall i = 1, s \quad \forall k = 1, M \quad b_{ik} = \Pr(Y_n = k / X_n = i) \quad (2.8)$$

$$\forall i = 1, s \quad \sum_{k=1}^M b_{ik} = 1 \quad (2.9)$$

(b_{ik} will also be denoted $b_i(k)$.)

The HMM λ is summarized by $\lambda = (\pi_1, A, B)$. The matrices π_1 , A , and B , whose entries are in $[0, 1]$, and whose elements sum to 1 in each of their rows, are *stochastic matrices*. All the probabilities of characteristic sequences can be expressed in terms of the model parameters λ . The probability of a state sequence $x(1:L)$ is given by:

$$\Pr[x(1:L)] = \prod_{n=1}^L a_{x_{n-1}x_n}, \quad (2.10)$$

where, by convention, $a_{x_0x_1} = a_{x_1}$. The conditional probability of an observation sequence $y(1:L)$ is given by:

$$\Pr[y(1:L) / x(1:L)] = \prod_{n=1}^L b_{x_n}(y_n). \quad (2.11)$$

The joint probability of the sequences x and y is given by:

$$\begin{aligned} \Pr(x, y) &= \Pr(y/x) \Pr(x) \\ \Pr(x, y) &= \prod_{n=1}^L a_{x_{n-1}x_n} b_{x_n}(y_n). \end{aligned} \quad (2.12)$$

This joint probability is also called the likelihood function. The log-likelihood $\mathbf{L}_L(x, y) = -\log_{10}[\Pr(x, y)]$ will be used in chapter 4. We have:

$$\mathbf{L}_L(x, y) = - \sum_{n=1}^L [\hat{a}_{x_{n-1}x_n} + \hat{b}_{x_n}(y_n)], \quad (2.13)$$

where the convention $\hat{a} = \log_{10} a$ has been used. Another log-likelihood, \mathcal{L} , based on all the possible state sequences, will be introduced in chapter 3.

2.2.2 Graphical representations

Two graphical representations of an HMM will be useful:

- the Markov chain diagrams: one to visualize the configuration of the model, and one to visualize the evolution of the system over time (see figures 2.1 and 2.2).
- the trellis structure: to formulate HMM problems as a search through a graph (see figure 2.3).

In the case of the trellis structure, the nodes (the states of the trellis) might actually represent speech-states or speech-observations. The observed sequence of data, used on top of the trellis, will be called the sequence of observations (even though these “observations” might actually be speech-states).

2.2.3 The steady-state distribution of the states

Under the proper conditions, a Markov chain with a finite number of states can be shown to have a stationary distribution of the states. In other words, there is a distribution $\pi = (\pi_i)_{i=1,s}$ such that:

$$\forall i = 1, s \quad \lim_{n \rightarrow \infty} \Pr(X_n = i) = \pi_i. \quad (2.14)$$

The distribution π is also called the steady-state distribution of the states. Regardless of the initial distribution of the states, the distribution of X_n approaches π as $n \rightarrow \infty$. The conditions for the existence and uniqueness of the steady-state distribution are discussed in [66,5]. The steady-state distribution of the states of

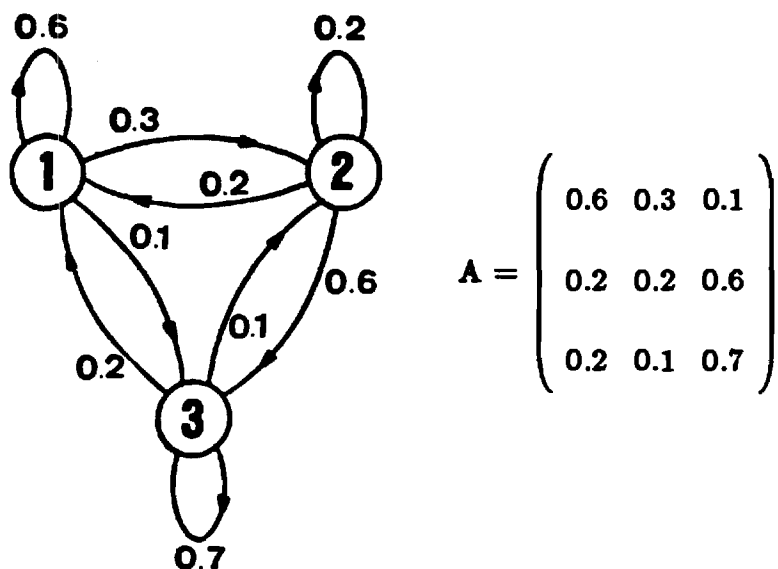


Figure 2.1: State diagram of a 3-state Markov Chain.

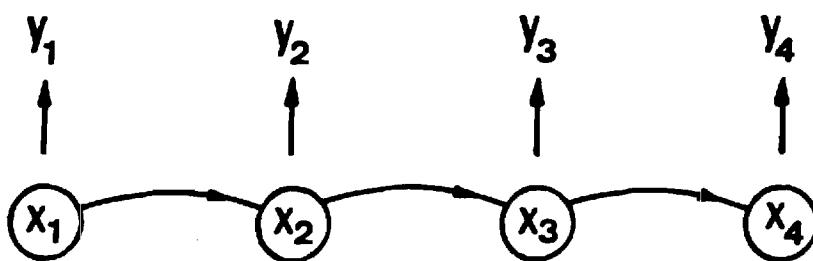


Figure 2.2: Evolution of a HMM system over time.

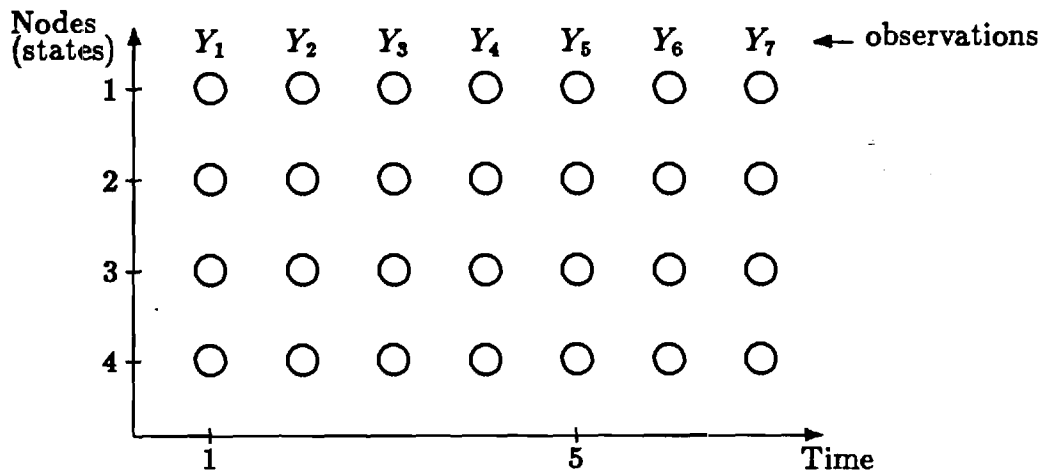


Figure 2.3: An HMM trellis.

a finite Markov chain can be found by solving a set of s linear equations with s unknowns. It is easily seen that:

$$\begin{aligned} \forall j = 1, s \quad \pi_j &= \sum_{i=1}^s a_{ij} \pi_i \quad \text{i.e.,} \\ \forall j = 1, s \quad 0 &= \sum_{i=1}^s (a_{ij} - \delta_{ij}) \pi_i \\ &(\delta_{ij} = 1 \text{ if } i = j, 0 \text{ otherwise}). \end{aligned}$$

$s - 1$ of these equations, and the stochastic constraints on π , constitute the linear system of equations used to solve for π :

$$\begin{cases} \sum_{i=1}^s (a_{ij} - \delta_{ij}) \pi_i = 0 & j = 1, s - 1 \\ \sum_{i=1}^s \pi_i = 1 \end{cases} \quad (2.15)$$

The distribution of the states, at any time n , can be approximated by the steady-state distribution of the states, or computed recursively from the initial distribution of the states with:

$$\begin{cases} \Pr(X_1 = i) = a_i \\ \Pr(X_n = j) = \sum_{i=1}^s \Pr(X_{n-1} = i) a_{ij} \end{cases} \quad (2.16)$$

2.3 The adjoined HMM

2.3.1 The model parameters

In the (direct) HMM $\lambda = (\pi_1, A, B)$, the observation Y_n is seen as directly dependent on X_n , and the state X_n is seen as directly dependent on X_{n-1} . The dependency of Y_n on Y_{n-1} is then indirectly modelled through the state sequence. The process can now be “reversed” by considering Y_n as directly dependent on Y_{n-1} , and X_n as directly dependent on Y_n . Thus we define an adjoined (or dual)² HMM, $\tilde{\lambda} = (\tilde{\pi}_1, \tilde{A}, \tilde{B})$:

$$\tilde{\pi}_1 = (\tilde{a}_k)_{k=1,M} \quad \tilde{a}_k = \Pr(Y_1 = k) \quad (2.17)$$

$$\tilde{A} = (\tilde{a}_{k\ell})_{\substack{k=1,M \\ \ell=1,M}} \quad \forall n \quad \tilde{a}_{k\ell} = \Pr(Y_n = \ell / Y_{n-1} = k) \quad (2.18)$$

$$\tilde{B} = (\tilde{b}_{kj})_{\substack{k=1,M \\ j=1,S}} \quad \forall n \quad \tilde{b}_{kj} = \Pr(X_n = j / Y_n = k). \quad (2.19)$$

$\tilde{\pi}_1$, \tilde{A} , and \tilde{B} are the dual stochastic matrices. $\tilde{\pi}_1$ is the initial distribution of the observations. \tilde{A} is the observation transition probability matrix. \tilde{B} is the state output probability matrix. The steady-state distribution of the observations is defined by³:

$$\tilde{\pi} = (\tilde{\pi}_k)_{k=1,M} \quad \forall n \quad \tilde{\pi}_k = \Pr(Y_n = k). \quad (2.20)$$

The probabilities $\Pr(Y_1)$, $\Pr(Y_n)$, $\Pr(X_n/Y_n)$, and $\Pr(Y_n/Y_{n-1})$ must be estimated. A direct practical estimation is usually difficult: $10M^2$ frames of speech are necessary to estimate $\Pr(Y_n/Y_{n-1})$ (on a basis of 10 occurrences per parameter), i.e., roughly 44 hours of speech (15ms frames) for $M = 1024$. However, an indirect estimation of the adjoined model $\tilde{\lambda}$, from the direct model λ , is possible:

- $\Pr(Y_1)$ can be estimated from a training sequence $Y(1:L)$, or this marginal

²An inverse HMM λ^{-1} will also be defined in chapter 4.

³Actually, $\tilde{\pi}_k = \lim_{n \rightarrow \infty} \Pr(Y_n = k)$ (when it exists) and $\Pr(Y_n = k)$ can be approximated by $\tilde{\pi}_k$.

probability can be computed as the sum of joint probabilities:

$$\begin{aligned}
 \Pr(Y_1) &= \sum_{X_1} \Pr(X_1, Y_1) = \sum_{X_1} \Pr(X_1) \Pr(Y_1/X_1) \\
 \Pr(Y_1 = k) &= \sum_{j=1}^s \Pr(X_1 = j) \Pr(Y_1 = k/X_1 = j) \quad \text{i.e.,} \\
 \tilde{a}_k &= \sum_{j=1}^s a_j b_{jk}. \tag{2.21}
 \end{aligned}$$

Note that, naturally, the stochastic constraint is implicitly satisfied:

$$\sum_{k=1}^M \tilde{a}_k = \sum_{k=1}^M \left[\sum_{j=1}^s a_j b_{jk} \right] = \sum_{j=1}^s a_j \left[\sum_{k=1}^M b_{jk} \right] = \sum_{j=1}^s a_j = 1.$$

• $\Pr(Y_n)$ can be computed similarly, from a training sequence, or from the steady-state distribution of the states:

$$\begin{aligned}
 \Pr(Y_n) &= \sum_{X_n} \Pr(X_n, Y_n) = \sum_{X_n} \Pr(X_n) \Pr(Y_n/X_n) \\
 \Pr(Y_n = k) &= \sum_{j=1}^s \Pr(X_n = j) \Pr(Y_n = k/X_n = j) \quad \text{i.e.,} \\
 \tilde{\pi}_k &= \sum_{j=1}^s \pi_j b_{jk}. \tag{2.22}
 \end{aligned}$$

The stochastic constraint $\sum_{k=1}^M \tilde{\pi}_k = 1$ is similarly satisfied.

• $\Pr(X_n/Y_n)$ could be directly estimated from the training sequence $Y(1:L)$, once the state sequence $X(1:L)$ would have been decoded. These probabilities can be more easily derived with Bayes' rule:

$$\begin{aligned}
 \Pr(X_n/Y_n) &= \frac{\Pr(X_n)}{\Pr(Y_n)} \Pr(Y_n/X_n) \\
 \Pr(X_n = j/Y_n = k) &= \frac{\Pr(X_n = j)}{\Pr(Y_n = k)} \Pr(Y_n = k/X_n = j) \quad \text{i.e.,} \\
 \tilde{b}_{kj} &= \frac{\pi_j}{\tilde{\pi}_k} b_{jk}. \tag{2.23}
 \end{aligned}$$

Note that $\sum_{j=1}^s \tilde{b}_{kj} = \frac{1}{\tilde{\pi}_k} \sum_{j=1}^s \pi_j b_{jk} = 1$.

• $\Pr(Y_n/Y_{n-1})$ could be directly estimated from the training sequence $Y(1:L)$. Unfortunately it would require a training sequence prohibitively long. However, they can be derived as the sum of joint probabilities:

$$\begin{aligned}\Pr(Y_n/Y_{n-1}) &= \sum_{X_{n-1}} \sum_{X_n} \Pr(Y_n, X_n, X_{n-1}/Y_{n-1}) \\ &= \sum_{X_{n-1}} \sum_{X_n} \Pr(X_{n-1}/Y_{n-1}) \Pr(Y_n, X_n/Y_{n-1}, X_{n-1}).\end{aligned}$$

The term $\Pr(Y_n, X_n/Y_{n-1}, X_{n-1})$ cannot be expanded as the product $\Pr(Y_n/Y_{n-1}, X_{n-1}) \Pr(X_n/Y_{n-1}, X_{n-1})$ because X_n and Y_n are dependent random variables. However, Bayes' rule can be used:

$$\Pr(Y_n, X_n/Y_{n-1}, X_{n-1}) = \Pr(X_n/Y_{n-1}, X_{n-1}) \Pr(Y_n/Y_{n-1}, X_{n-1}, X_n).$$

The assumptions behind an HMM, defined in section 2.1 on stochastic modelling, can now be used:

$$\begin{aligned}\Pr(X_n/Y_{n-1}, X_{n-1}) &= \Pr(X_n/X_{n-1}) \\ \Pr(Y_n/Y_{n-1}, X_{n-1}, X_n) &= \Pr(Y_n/X_n).\end{aligned}$$

Therefore $\Pr(Y_n, X_n/Y_{n-1}, X_{n-1}) = \Pr(X_n/X_{n-1}) \Pr(Y_n/X_n)$ and:

$$\begin{aligned}\Pr(Y_n/Y_{n-1}) &= \sum_{X_{n-1}} \sum_{X_n} \Pr(X_{n-1}/Y_{n-1}) \Pr(X_n/X_{n-1}) \Pr(Y_n/X_n) \quad \text{or:} \\ \Pr(Y_n = \ell/Y_{n-1} = k) &= \sum_{i=1}^I \sum_{j=1}^I [\Pr(X_{n-1} = i/Y_{n-1} = k) \Pr(X_n = j/X_{n-1} = i) \\ &\quad \Pr(Y_n = \ell/X_n = j)] \quad \text{i.e.:}\end{aligned}$$

$$\tilde{a}_{k\ell} = \sum_{i=1}^I \sum_{j=1}^I \tilde{b}_{ki} a_{ij} b_{j\ell}. \quad (2.24)$$

Note that:

$$\sum_{\ell=1}^M \tilde{a}_{k\ell} = \sum_{i=1}^I \sum_{j=1}^I \tilde{b}_{ki} a_{ij} \left[\sum_{\ell=1}^M b_{j\ell} \right] = \sum_{i=1}^I \tilde{b}_{ki} \left[\sum_{j=1}^I a_{ij} \right] = \sum_{i=1}^I \tilde{b}_{ki} = 1.$$

The previous relations between $\tilde{\lambda}$ and λ can be rewritten as matrix equations.

Let us define two diagonal matrices:

- an $M \times M$ matrix, $D_1 = \text{diag}(\frac{1}{\tilde{\pi}_k})_{k=1,M}$:

$$D_1 = \begin{pmatrix} \frac{1}{\tilde{\pi}_1} & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{\tilde{\pi}_2} & 0 & \dots & 0 \\ 0 & 0 & \ddots & \vdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \frac{1}{\tilde{\pi}_M} \end{pmatrix} \quad (2.25)$$

- an $s \times s$ matrix, $D_2 = \text{diag}(\pi_j)_{j=1,s}$:

$$D_2 = \begin{pmatrix} \pi_1 & 0 & 0 & \dots & 0 \\ 0 & \pi_2 & 0 & \dots & 0 \\ 0 & 0 & \ddots & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \pi_s \end{pmatrix} \quad (2.26)$$

The equations (2.21), (2.22), (2.23), and (2.24) can now be expressed by:

$$\tilde{\pi}_1 = \pi_1^T B \quad (2.27)$$

$$\tilde{\pi} = \pi^T B \quad (2.28)$$

$$\tilde{B} = D_1 B^T D_2 \quad (2.29)$$

$$\tilde{A} = \tilde{B} A B \quad (2.30)$$

2.3.2 Likelihood and distance measure

The likelihood previously defined in equation (2.12) can now be expressed in terms of the parameters $\tilde{\lambda}$:

$$\begin{aligned} \Pr(x, y) &= \Pr(x/y) \Pr(y) \\ \Pr[y(1:L)] &= \prod_{n=1}^L \tilde{a}_{y_{n-1}y_n} \\ \Pr[x(1:L)/y(1:L)] &= \prod_{n=1}^L \tilde{b}_{y_n}(x_n) \\ \Pr(x, y) &= \prod_{n=1}^L \tilde{a}_{y_{n-1}y_n} \tilde{b}_{y_n}(x_n) \end{aligned} \quad (2.31)$$

$$\Pr(x, y) = \prod_{n=1}^L \frac{\pi_{x_n}}{\tilde{\pi}_{y_n}} b_{x_n}(y_n) \Pr(y_n/y_{n-1}). \quad (2.32)$$

The parameters of the adjoined model $\tilde{\lambda}$ can be directly used in equation (2.31). Each term in the product depends only on the present and previous observation, and on the current state. In theory, the likelihood defined by equation (2.31) is the same as the one defined by equation (2.12). In practice however, different estimates of $\Pr(y_n/y_{n-1})$ can be used in equation (2.32). For example, the probability derived from the LPC log-likelihood ratio, or the Itakura-Saito distance measure, could be used. Inversely, the probability $\Pr(y_n/y_{n-1})$ derived from the adjoined model can be used to define a new distance measure $D(y_{n-1}, y_n)$, for example:

$$D(y_{n-1}, y_n) = \exp[-\Pr(y_{n-1}/y_n)] \quad \text{i.e.,} \quad (2.33)$$

$$D(y_{n-1}, y_n) = \exp[-\tilde{a}_{y_{n-1}y_n}]. \quad (2.34)$$

This distance measure is different from classical speech distance measures because the time dimension is included. The distance, not only gives a measure of how close two observations are, but also provides a measure of how closely the next observation should follow the current observation in the sequence. The measure of closeness is not based anymore on an absolute comparison between observations, but on a relative comparison between two observations, as a function of their

relative position in the sequence. Even though $D(y_{n-1}, y_n)$ was initially defined for discrete observations, it could also be fitted to a multidimensional function of continuous observations. Another approach could be to map the multidimensional observations Y_i 's (for example 10 dimensional LPC vectors), onto a 2 dimensional Euclidean space, using multidimensional scaling [84], with $D(y_{n-1}, y_n)$ as a similarity measure.

2.4 Entropy, bit rate, and structure

2.4.1 The concept of entropy

The concept of entropy has been used in several branches of science and engineering, such as physics, chemistry, and thermodynamics. We will present here an elementary discussion of the concept of entropy from the information theory point of view [5,57], for discrete memoryless sources⁴. A more general discussion can be found in [57]. Consider a random variable Z which takes values from the source alphabet $\mathcal{A}_S = \{1, 2, \dots, N\}$ with the respective probabilities $\{\text{Pr}(1), \text{Pr}(2), \dots, \text{Pr}(N)\}$. This is also referred to as the source Z producing source letters from the alphabet \mathcal{A}_S . These source letters are encoded with fixed or variable length codewords, built from code letters drawn from a code alphabet \mathcal{A}_C of D symbols. For a binary code, $D = 2$, and the code letters are the bits 0 and 1. The entropy per letter of the source Z is given by:

$$H(Z) = - \sum_{i=1}^N \text{Pr}(i) \log_2 \text{Pr}(i). \quad (2.35)$$

The entropy is the expected value of the random variable $\hat{Z} = -\log_2 \text{Pr}(Z)$. According to the *source coding theorem* [57], the average number of code letters per source letter, \bar{n} , necessary to perfectly encode blocks of K source letters satisfies:

$$\frac{H(Z)}{\log_2 D} \leq \bar{n} < \frac{H(Z)}{\log_2 D} + \frac{1}{K}. \quad (2.36)$$

⁴Successive source letters are statistically independent.

In the case of a binary code ($D = 2$), we have:

$$H(Z) \leq \bar{n} < H(Z) + \frac{1}{K}, \quad (2.37)$$

therefore the entropy is the minimum number of bits per source letter, necessary to represent the source Z . When K is large, $H(Z)$ is *the average number of bits per source letter* necessary to encode the source perfectly. Entropy coding schemes, such as Huffman coding [68], allow encoding of the source Z with a bit rate equal to the one predicted by the entropy $H(Z)$. Since the number of bits of information necessary to encode a source reflects the structure of the source, the entropy is also a measure of the structure and amount of order that can be expected in a sequence $Z(1:L)$. The lower the entropy, the more structure there is into the system, and the less number of bits of information are necessary to represent it. In that respect, the entropy is bounded as follows:

$$0 \leq H(Z) \leq \log_2 N \quad (2.38)$$

The lower bound of the entropy is reached for a totally ordered system, with no uncertainty and total predictability. We have total a priori information on the state of the system, and its structure is perfectly known. This is obtained if:

$$\exists i_0 \in [1, N] \text{ s.t. } \Pr(Z = i_0) = 1 \quad \text{and} \quad \forall i \neq i_0 \Pr(Z = i) = 0.$$

$H(Z) = 0$, and the system is “deterministically” known to be in state i_0 . The upper bound of the entropy is reached for a maximally disordered system, with total uncertainty and no predictability. There is no a priori information about the state of the system and no apparent structure. This is obtained if:

$$\forall i \in [1, N] \quad \Pr(Z = i) = \frac{1}{N}.$$

$H(Z) = \log_2 N$, and the N source letters are equiprobable.

2.4.2 Entropies of an HMM

The concept of entropy for discrete memoryless sources can be extended to discrete non-memoryless (dependent) sources, in particular to hidden Markov models⁵. Even though the state and observation random variables X_t and Y_t are dependent sources, they could be approximated by memoryless sources, and the entropy $H(Z)$ (where $Z = X$ or $Z = Y$) could be computed. However, these entropies would not generally be very interesting because they would ignore the inherent dependencies and structures modelled by the HMM. Anyhow the entropy $H(Z)$ can be computed for the steady-state distribution of the states π , and observations $\tilde{\pi}$. For example:

$$H(\pi) = - \sum_{i=1}^I \pi_i \log_2 \pi_i. \quad (2.39)$$

Outside from these entropies, other entropies directly taking into account the structure of the HMM, should be defined: the state entropy $H(X)$, and the observation entropy $H(Y)$. Basically these entropies operate on an alphabet of letters made of pairs such as (X_{n-1}, X_n) and (X_n, Y_n) . First we define conditional entropies:

- $H(X_n/X_{n-1} = i)$ the entropy of the state source X_n conditioned on the value of the previous state X_{n-1} — i.e., an entropy defined on the pair (X_{n-1}, X_n) or the transition $X_{n-1} \rightarrow X_n$.
- $H(Y_n/X_n = j)$ the entropy of the observation source Y_n conditioned on the value of the present state X_n — i.e., an entropy defined on the pair (X_n, Y_n) .

Contrary to $H(Z)$ which was a number, $H(X_n/X_{n-1})$ and $H(Y_n/X_n)$ are random variables, whose outcomes depend on the outcomes of X_{n-1} and X_n respectively. These entropies are given by:

$$H(X_n/X_{n-1} = i) = - \sum_{j=1}^I a_{ij} \log_2 a_{ij} \quad (2.40)$$

$$H(Y_n/X_n = j) = - \sum_{k=1}^M b_{jk} \log_2 b_{jk}. \quad (2.41)$$

⁵Very similar to the concept of Markov sources in information theory.

The interpretation of these two entropies is similar to that of $H(Z)$ for fixed $X_{n-1} = i$ or $X_n = j$. These two random variables have expected values, the desired state and observation entropies $H(X)$ and $H(Y)$:

$$H(X) = \langle H(X_n/X_{n-1} = i) \rangle \quad (2.42)$$

$$H(Y) = \langle H(Y_n/X_n = j) \rangle \quad \text{i.e.,} \quad (2.43)$$

$$H(X) = \sum_{i=1}^s \pi_i H(X_n/X_{n-1} = i) \quad (2.44)$$

$$H(Y) = \sum_{j=1}^M \pi_j H(Y_n/X_n = j) \quad \text{or,} \quad (2.45)$$

$$H(X) = - \sum_{i=1}^s \sum_{j=1}^M \pi_i a_{ij} \log_2 a_{ij} \quad (2.46)$$

$$H(Y) = - \sum_{j=1}^M \sum_{k=1}^M \pi_j b_{jk} \log_2 b_{jk}. \quad (2.47)$$

The results concerning bit rate and system structure, presented above for $H(Z)$, still hold for $H(X)$ and $H(Y)$. In particular the source coding theorem applies to the observation source Y . As a consequence, the following interpretations of the entropies of an HMM can be formulated: for the state source X , $H(X_n/X_{n-1} = i)$ is the number of bits required to encode X_n , given $X_{n-1} = i$. There are roughly $2^{H(X_n/X_{n-1}=i)}$ plausible transitions from $X_{n-1} = i$ to X_n (i.e., $2^{H(X_n/X_{n-1}=i)}$ plausible states X_n following state $X_{n-1} = i$). Then $H(X_n/X_{n-1} = i)$ is averaged over all possible previous states X_{n-1} . The same reasoning holds for the observation source. Therefore, as a summary:

- $2^{H(X)}$ represents the average number of plausible transitions, at a given time, from a given state. This number would be 1 for a totally structured system, and s for a totally random system.
- $2^{H(Y)}$ represents the average number of plausible observations likely to be produced, at a given time, by a given state. This number would be 1 for a totally structured system, and M for a totally random system.

To encode the state and observation sources, s different Huffman codes (one respectively associated with each $X_{n-1} = i$ or $X_n = j$) can be used. The previous state X_{n-1} (resp. the current state X_n) identifies which code to use to decode the current state X_n (resp. the current observation Y_n). The entropies are bounded as follows:

$$0 \leq H(X) \leq \log_2 s \quad (2.48)$$

$$0 \leq H(Y) \leq \log_2 M. \quad (2.49)$$

For an HMM with $s = 64$, and $M = 1024$ we have:

$$0 \leq H(X) \leq 6 \quad \text{and} \quad 0 \leq H(Y) \leq 10.$$

The lower bounds of the entropies are reached for *diagonal* matrices, independently of the value of the distribution π . The upper bounds are reached for “*equiprobable*” matrices ($\forall i, j, k \ a_{ij} = 1/s, \ b_{jk} = 1/M$) independently of the value of the distribution π . The state entropy $H(X)$ will also be called the entropy of the transition matrix A , and sometimes denoted by H_A . The observation entropy $H(Y)$ will sometimes be called the entropy of the observation probability matrix B , and will also be denoted by H_B . These entropies are two particular examples of the entropy of a (rectangular) stochastic matrix. In figures 2.4 and 2.5, we computed the entropy (to be compared to $0 \leq H(X) \leq 2$) of the transition matrix of respectively a loosely structured and a highly structured 4-state Markov chain.

2.5 Fundamental issues of HMM's

So far we described an HMM, together with the methods to derive its main characteristics, when the model λ is known. Now we should be concerned with the following important issues of a hidden Markov model: how to find it, how to use it, how to interpret it?

$$A = \begin{pmatrix} 0.40 & 0.22 & 0.18 & 0.20 \\ 0.20 & 0.35 & 0.15 & 0.30 \\ 0.17 & 0.23 & 0.26 & 0.34 \\ 0.01 & 0.15 & 0.35 & 0.49 \end{pmatrix} \Rightarrow \pi = \begin{pmatrix} 0.154 \\ 0.227 \\ 0.255 \\ 0.364 \end{pmatrix} \quad H_A = 1.782$$

Figure 2.4: State entropy of a loosely structured 4-state Markov chain.

$$A = \begin{pmatrix} 0.60 & 0.40 & 0.00 & 0.00 \\ 0.00 & 0.56 & 0.44 & 0.00 \\ 0.00 & 0.00 & 0.67 & 0.33 \\ 0.00 & 0.00 & 0.50 & 0.50 \end{pmatrix} \Rightarrow \pi = \begin{pmatrix} 0.000 \\ 0.000 \\ 0.602 \\ 0.398 \end{pmatrix} \quad H_A = 0.949$$

Figure 2.5: State entropy of a highly structured 4-state Markov chain.

An HMM is defined by the parameters $\lambda = (\pi_1, A, B)$. How can these parameters be estimated? We would like them to be estimated automatically (by a computer program) from a finite observed sequence $Y(1:L)$. The process of deriving (learning) the model probabilities automatically from a finite speech corpus is called the *training process*. The problem will be addressed in chapter 3.

Now that the model probabilities are known, how can they be used to process an input signal? Two inverse processes need to be addressed: the *analysis phase* and the *synthesis phase*. In the analysis phase, the problem is to recover the hidden state structure of the speech from its observed waveform. Given a speech observation Y_t , what is the state which generated it? In these terms however, the problem is not correctly formulated, because speech is a global process with dependencies between its elements, as modelled by an HMM. Therefore the right question would be: given an observed speech sequence $Y(1:L)$, what state sequence $X(1:L)$ is more likely to have produced it? Solving this problem is also known as performing the *state decoding*. Similarly and inversely, in the synthesis phase, the problem is to synthesize intelligible speech from a known deep state structure. An answer to the following question should be provided: given a state sequence $X(1:L)$, what is the most likely and intelligible speech sequence $Y(1:L)$ it generates? Synthesizing speech from the states is known as performing the *observation decoding*. These two problems will be addressed in chapter 4.

Finally we would like to correlate the abstract deep state structure of the speech with more intuitive and classical speech representations — for example, as given by linguistics: phonetic descriptions etc. This is known as the *state interpretation* problem. Even though a state interpretation is not necessary for the training, analysis, and synthesis phases (a strength of the modelling!), it would provide more insight into the inner workings of the modelling, and make the synthesis phase easier. It would also be necessary for the application of this model to

automatic speech recognition.

2.6 Other types of HMM's

2.6.1 Continuous hidden Markov models [78,91,122,123]

So far we described discrete hidden Markov models (DHMM's)⁶. In this case the observations could take only a finite number of discrete values — the observations were drawn from a codebook of size M . It is also possible to define a continuous HMM (CHMM) for which observations can take an infinite number of values from a continuous support — like the real Euclidean space \mathbf{R}^P . To describe CHMM's, the observation probabilities of a DHMM, b_{jk} , need to be replaced by probability density functions $b_j(Y)$. Some of the most common density functions used with CHMM's, for P dimensional vectors Y 's, are described below⁷:

- The Normal (or Gaussian) distribution:

$$\mathcal{N}(Y, \mu, C) = \frac{1}{(2\pi)^{P/2} |C|^{1/2}} \exp\left[-\frac{1}{2}(Y - \mu)^T C^{-1}(Y - \mu)\right], \quad (2.50)$$

where μ is the mean vector, and C the covariance matrix. The output density functions are:

$$b_j(Y) = \mathcal{N}(Y, \mu_j, C_j). \quad (2.51)$$

- The Normal distribution with a diagonal covariance matrix $C = \text{diag}(\sigma_i^2)_{i=1,s}$:

$$\mathcal{N}_d(Y, \mu, C) = \frac{1}{(2\pi)^{P/2} (\prod_{i=1}^P \sigma_i^2)^{1/2}} \exp\left[-\frac{1}{2} \sum_{i=1}^P \frac{(Y_i - \mu_i)^2}{\sigma_i^2}\right] \quad (2.52)$$

i.e.,

$$\mathcal{N}_d(Y, \mu, C) = \prod_{i=1}^P \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left[-\frac{(Y_i - \mu_i)^2}{2\sigma_i^2}\right] = \prod_{i=1}^P \mathcal{N}(Y_i, \mu_i, \sigma_i), \quad (2.53)$$

⁶It stands for discrete observation HMM's.

⁷Actually, for DHMM's, Y_t represented the index of the observation in the codebook; now, for CHMM's, it represents the actual P dimensional observation vector, and will sometimes be denoted O_t to avoid possible confusion (see chapter 3).

therefore

$$b_j(Y) = \prod_{i=1}^P \mathcal{N}(Y_i, \mu_{ji}, \sigma_{ji}). \quad (2.54)$$

Even though unimodal probability density functions, like the multivariate Gaussian, might be appropriate for some multi-dimensional observations Y 's (like mel-cepstrum coefficients), usually multimodal densities are needed to accurately describe most of the multi-dimensional speech observations (like prediction coefficients). A mixture of normal densities can approximate any probability density function with any desirable accuracy, as long as the number of mixtures G is allowed to be as large as necessary. The output densities are given by:

- The Gaussian mixture density:

$$b_j(Y) = \sum_{m=1}^G \lambda_{jm} \mathcal{N}(Y, \mu_{jm}, C_{jm}) \quad (2.55)$$

with: $\forall j = 1, s \quad \sum_{m=1}^G \lambda_{jm} = 1.$

2.6.2 Hidden semi-Markov models [66,87,88]

For an HMM, the state duration distributions are exponentially decreasing with time, and reach their maximum for the shortest duration $n = 1$. The probability of staying n units of time in state i is:

$$\Pr(D = n/i) = (1 - a_{ii})a_{ii}^{n-1}. \quad (2.56)$$

Hidden semi-Markov models (HSMM's)⁸ are used to allow other state duration distributions, that model the physical phenomenon under study more closely. An HSMM is represented by a transition matrix with zeros along its diagonal. Once a state i has been reached, the number of self-transitions $i \rightarrow i$ is determined by the state duration distribution. The transition to a different state $i \rightarrow j$ is governed by

⁸They are also called pure jump processes.

the transition matrix. An HSMM is uniquely defined by $\eta = (\pi_1, A, B, D)$ where⁹:

$$D = (d_i(n))_{i=1,s} \quad d_i(n) = \Pr(D_i = n / X_i = i). \quad (2.57)$$

$d_i(n)$ is simply the probability that the system will stay n units of time in state i , when i is reached for the first time. Usually parametric state duration distributions are used to limit the number of parameters to estimate. Common examples are [110, p.103]:

- The Poisson distribution ($\lambda_i > 0$):

$$d_i(n) = e^{-\lambda_i} \frac{\lambda_i^n}{n!} \quad (2.58)$$

- The Gamma distribution ($\lambda_i > 0, \eta_i > 0$):

$$d_i(n) = A_i n^{\eta_i} e^{-\lambda_i n} \quad (2.59)$$

$$A_i = \frac{\lambda_i^{\eta_i+1}}{\Gamma(\eta_i + 1)}. \quad (2.60)$$

⁹In this simple formulation, t represents every first time a new state is reached. Stationarity makes the duration probability independent of t .

CHAPTER 3

Maximum likelihood estimation of the parameters of hidden Markov models

3.1 Presentation of the problem

A first order, stationary discrete hidden Markov model (DHMM), as described in chapter 2, is uniquely defined by the following 3 parameters:

- the initial distribution of the states $\pi_1 = (a_i) \quad i = 1, s$:

$$a_i = \Pr(X_1 = i) \quad (3.1)$$

- the transition probability matrix $A = (a_{ij}) \quad i = 1, s \quad j = 1, s$:

$$\forall n \quad a_{ij} = \Pr(X_{n+1} = j / X_n = i) \quad (3.2)$$

- the output probability matrix $B = (b_{ik}) \quad i = 1, s \quad k = 1, M$:

$$\forall n \quad b_{ik} = \Pr(Y_n = k / X_n = i) \quad (3.3)$$

(b_{ik} will also be denoted $b_i(k)$)

Representing a physical system by a DHMM means estimating the model parameters $\lambda = (\pi_1, A, B)$. The estimation process is also called the “training process” in the sense that the above probabilities have to be “automatically learned”

by the system. As we have previously seen, only the observations Y_n are directly accessible to experiment. The underlying states X_n are not directly accessible, but are hidden. The problem is then to estimate λ from a finite length observed sequence $Y(1:L) = \{Y_1, Y_2, \dots, Y_L\}$ known as the training sequence. If the nature of the states was identified, and if the state sequence $X(1:L)$ corresponding to a given observation sequence $Y(1:L)$ was known, then direct frequency counts could be used to estimate λ . If f_{ij} is the number of transitions from state i to state j in $X(1:L)$ and f_i the number of transitions out of state i in $X(1:L)$, then the maximum likelihood estimate of a_{ij} is given by:

$$a_{ij} = f_{ij} / f_i \quad (3.4)$$

(note that: $f_i = \sum_{t=1}^L f_{it}$).

If h_{ik} is the number of times the observation k , in $Y(1:L)$, is produced from state i , in $X(1:L)$, and h_i is the number of times state i occurs in $X(1:L)$, ($h_i = f_i$ or $h_i = f_i + 1$), then the maximum likelihood estimate of b_{ik} is given by:

$$b_{ik} = h_{ik} / h_i \quad (3.5)$$

(note that: $h_i = \sum_{k=1}^M h_{ik}$).

If only one training sequence is available, starting with state i_0 , then π_1 is estimated by:

$$a_i = \begin{cases} 1 & \text{if } i = i_0 \\ 0 & \text{otherwise.} \end{cases} \quad (3.6)$$

If N training sequences are available, let n_i be the number of sequences starting with state i , then π_1 is estimated by:

$$\forall i = 1, s \quad a_i = n_i / N. \quad (3.7)$$

Note that, if only one training sequence is available, a_i can still be computed: the stationary assumption allows to assimilate ensemble and time averages, i.e., to estimate the initial distribution of the states by the steady state distribution of the states (which can be directly derived from the transition matrix, as seen in chapter 2).

If the states were directly observable (i.e., if states and observations were the same) then the B matrix would be reduced to the identity matrix. Given that the states are unknown (in nature and as a sequence), a direct frequency count estimation is impossible. However one can consider all possible state sequences $X^{(\ell)}$ for a given observation sequence Y . There are s^L such sequences; each one of them is identified by the superscript (ℓ) . The previous frequency counts can be (theoretically) computed for each of these sequences. Then the model parameters can be estimated. For example, the transition matrix is evaluated by:

$$a_{ij} = \sum_{\ell=1}^{s^L} P(X^{(\ell)}) \frac{f_{ij}^{(\ell)}}{f_i^{(\ell)}} \quad (3.8)$$

where $P(X^{(\ell)})$ is the probability of $X^{(\ell)}(1:L)$ and $f_{ij}^{(\ell)}$, $f_i^{(\ell)}$ are the corresponding counts. This probability can be expressed by:

$$P(X^{(\ell)}) = \prod_{n=1}^L a_{x_{n-1}^{(\ell)} x_n^{(\ell)}}, \quad (3.9)$$

where by convention $a_{x_0^{(\ell)} x_1^{(\ell)}} = a_{x_1^{(\ell)}}$. Then from (3.8) we get a "reestimation relation" \mathcal{R} of the form:

$$a_{ij} = \mathcal{R} \left(\sum_{\ell=1}^{s^L} \prod_{n=1}^L a_{x_{n-1}^{(\ell)} x_n^{(\ell)}} \frac{f_{ij}^{(\ell)}}{f_i^{(\ell)}} \right). \quad (3.10)$$

Unfortunately, this approach does not solve the problem, at least for the following two reasons:

- the term a_{ij} appears in both the left and right-hand side of the equation, and there is no possible analytical solution.
- the huge number of state sequences (s^L) prevents any practical computation, even with the fastest possible computer.

However, the previous equation suggests an iterative procedure to estimate a_{ij} . The iteration process will depend on the general approach used. Mutual information maximization [11] and discrimination information minimization [48] approaches, Lagrangian techniques [86], and optimization procedures [111] are possible. The maximum likelihood estimation (MLE) approach, theoretically well-founded and practically efficient, will be the one described and used in this research.

The MLE can be formulated as follows: given a sequence Y of observations, estimate the parameters of the model to the ones which maximize the probability of producing the given Y . In other words $\lambda = (\pi_1, A, B)$ is a solution to the constrained maximization problem of the likelihood P :

$$\frac{\partial P}{\partial \lambda} = 0 \quad \text{subject to} \quad \sum_{j=1}^s a_{ij} = 1, \quad \sum_{i=1}^s a_i = 1, \quad \sum_{k=1}^M b_{ik} = 1 \quad (3.11)$$

(λ is meant to be all parameters of the model)

where:

$$P = \text{Pr}(Y/\lambda) \quad (3.12)$$

$$P = \sum_{\ell=1}^{s^L} P(Y, X^{(\ell)}) \quad (3.13)$$

$$P = \sum_{\ell=1}^{s^L} P(X^{(\ell)})P(Y/X^{(\ell)}) \quad (3.14)$$

with:

$$P(Y/X^{(\ell)}) = \prod_{n=1}^L b_{x_n^{(\ell)}}(Y_n) \quad (3.15)$$

and $P(X^{(\ell)})$ is given in (3.9).

P , referred to as the likelihood, is the sum over all possible state sequences of the weighted probability of the training sequence given a state sequence (the weighting factor being the probability of this state sequence). P is globally expressed by:

$$P = \sum_{\ell=1}^{s^L} \prod_{n=1}^L a_{x_{n-1}^{(\ell)} x_n^{(\ell)}} b_{x_n^{(\ell)}}(Y_n). \quad (3.16)$$

Because P can be very small, the log-likelihood $\mathcal{L} = -\log_{10} P$ will be often used. A solution λ_0 of (3.11) is a *critical point*. The direct evaluation of P requires Ls^L multiplications and additions. Since this factor is exponential with the training sequence length L , direct practical maximization is prevented, even for small models ($s = 2, M = 6, L = 100$), obviously for simple models ($s = 5, M = 64, L = 1000$), and most of all for realistic continuous speech models ($s = 64, M = 1024, L = 60000$). Therefore the problem must be dealt with in a considerably more efficient manner.

An iterative procedure, called the forward-backward algorithm (FBA) or Baum's algorithm, was developed by Baum et al. [15,16,17,18] to solve just this maximization problem.

3.2 The theory of the forward-backward algorithm

3.2.1 General results

The FBA can be summarized in terms of three main results:

- the maximum likelihood estimate of the model parameters λ converges to the

true value of λ as $L \rightarrow \infty$.

- there exists a growth transformation T which, applied to the model parameters λ , is sure to increase the likelihood P , except at a critical point of P (or equivalently at a fixed point of T); i.e., if C is the set of critical points of P then:

$$\forall \lambda \notin C \quad P(T\lambda) > P(\lambda). \quad (3.17)$$

Therefore the FBA starts with a random initial estimate λ_i and iteratively applies T . Convergence to a local maximum of P is guaranteed. The local maximum reached depends on the initial estimate λ_i (if C is not a singleton).

- the transformation T can be expressed analytically and evaluated recursively in terms of partial training sequences; the amount of computation required for P is linear in L (no longer exponential in L).

3.2.2 Historical development of the algorithm

- 1957: paper by Blackwell et al. [27]:

Two different HMM's λ_1 and λ_2 can generate the same finite length observation sequence Y . Now, inversely, suppose an HMM is to be estimated from the sequence Y . Is the estimation possible? What model will be recovered (λ_1 or λ_2 or another one)? What conditions on the training sequence should be met to recover a specific HMM? What parameters of the HMM are identifiable, which ones are not? This is referred to as the *identifiability problem* for functions of finite Markov chains. The paper "almost solves" the problem in two special cases.

- 1959: paper by Gilbert [59]:

The same problem is addressed and a more general solution is provided. A parametric representation of the equivalence class of all $s \times s$ transition matrices which give rise to the same distribution of observations is given.

- 1966: paper by Baum et al. [15]:

The proof that the MLE converges to the true value of λ is given. A central limit theorem for the observation process Y is presented.

- 1967: paper by Baum et al. [16]:

A proof of an inequality of the form $\mathcal{P}[\mathcal{T}(\lambda)] > \mathcal{P}(\lambda)$ is given, where \mathcal{P} is a polynomial and \mathcal{T} a defined transformation. The special application to the case where $\mathcal{P} = P$ (likelihood) and $\mathcal{T} = T$ (FBA growth transformation) is presented.

- 1970: paper by Baum et al. [17]:

A recurrent maximization technique for the MLE is presented. The analytical form of the growth transformation T is given. The proof of the convergence of the iterative procedure is based on the maximization of an auxiliary function Q with a unique global maximum. The algorithm applies to discrete probability mass functions $b_i(k)$, and strictly log-concave univariate continuous probability density functions $b_i(Y)$ — in particular the normal, Poisson, binomial, gamma, but not Cauchy, densities.

- 1972: paper by Baum [18]:

This summary paper presents the reestimation formulas for an HMM (i.e., the transformation T for a probabilistic function of a Markov chain) in terms of recursive computations based on forward and backward probabilities for partial sequences of the training sequence.

- 1982: paper by Liporace [91]:

The strictly log-concave class of continuous monovariate density functions is replaced by the more general class of elliptically symmetric multivariate densities, therefore broadening the scope of the FBA. Recursive reestimation formulas are given.

- 1983: paper by Levinson et al. [86]:

The FBA reestimation formulas are reviewed and practical implementation considerations for simple models and left-to-right models are presented.

- 1985: paper by Juang [78]:

The class of density functions studied by Liporace (see above) is extended to multivariate mixture densities, obviating the assumption of ellipsoidal symmetry. The general form of density functions now being allowed is a sum of strictly log-concave and/or elliptically symmetric densities.

3.2.3 Formulation and interpretation of the growth transformation

3.2.3.1 Partial derivatives formulation [18,86]

Although it neither constitutes a proof of the convergence to a local maximum of the FBA iterative procedure, nor provides a practically applicable solution, direct maximization of the likelihood P through a Lagrangian technique leads to reestimation formulas equivalent to Baum's formulas (after processing the partial derivatives with forward and backward probabilities). The solution of the likelihood equations (3.11) has the following form:

$$\bar{a}_{ij} = \frac{a_{ij} \partial P / \partial a_{ij}}{\sum_{t=1}^T a_{it} \partial P / \partial a_{it}} \quad (3.18)$$

$$\bar{b}_{ik} = \frac{b_{ik} \partial P / \partial b_{ik}}{\sum_{t=1}^T b_{it} \partial P / \partial b_{it}} \quad (3.19)$$

$$\bar{a}_i = \frac{a_i \partial P / \partial a_i}{\sum_{t=1}^T a_t \partial P / \partial a_t} \quad (3.20)$$

From these expressions it follows that any 0 parameter will be reestimated to 0. Any matrix with only one non-zero element in each of its rows — this element necessarily being 1 to satisfy the constraints (3.11) — will be reevaluated to itself. This holds in particular for a diagonal matrix (the identity matrix). This feature

of the reestimation formulas also allows to constrain the model to a specific given structure by setting the desired entries to 0 in the initial estimates. As an example, left-to-right models¹ are generated using an upper triangular matrix ($a_{ij} = 0$ if $j < i$) for the initial estimate of the transition matrix A .

3.2.3.2 Expected frequencies formulation [18,86]

Let $f_{ij}(\ell)$ (resp. $h_{ik}(\ell)$) be the f_{ij} (resp. h_{ik}) previously defined in section 3.1 for the state sequence $X^{(\ell)}$. In other words:

- $f_{ij}(\ell)$ is the number of transitions from state i to j in $X^{(\ell)}$.
- $h_{ik}(\ell)$ is the number of times the observation k from Y is generated by state i from $X^{(\ell)}$.

Let:

$$\delta_i(\ell) = \begin{cases} 1 & \text{if } X^{(\ell)} \text{ starts with state } i \\ 0 & \text{otherwise.} \end{cases}$$

Let n_{ij} , m_{ik} , q_i be the respective expected values of f_{ij} , h_{ik} and δ_i based on model λ , i.e.,

$$n_{ij} = \sum_{\ell=1}^L P(X^{(\ell)}, Y/\lambda) f_{ij}(\ell) \quad (3.21)$$

$$m_{ik} = \sum_{\ell=1}^L P(X^{(\ell)}, Y/\lambda) h_{ik}(\ell) \quad (3.22)$$

$$q_i = \sum_{\ell=1}^L P(X^{(\ell)}, Y/\lambda) \delta_i(\ell). \quad (3.23)$$

Then the reestimation formulas can be expressed by:

¹For a left-to-right model, the transition from state i to state j is allowed if and only if $j \geq i$, i.e., the system can jump only to higher order states, and a state which has been left cannot be reached again.

$$\bar{a}_{ij} = \frac{n_{ij}}{\sum_{\ell=1}^s n_{i\ell}} \quad (3.24)$$

$$\bar{b}_{ik} = \frac{m_{ik}}{\sum_{\ell=1}^s m_{i\ell}} \quad (3.25)$$

$$\bar{a}_i = \frac{q_i}{\sum_{\ell=1}^s q_{\ell}}. \quad (3.26)$$

In the case of a diagonal B matrix, only the state sequence $X^{(\ell_0)} = Y$ is possible:

$$\forall \ell \neq \ell_0 \quad P(X^{(\ell)}, Y/\lambda) = 0,$$

therefore the reestimation formulas can exactly be reduced to the frequency counts formulas presented in section 3.1.

3.2.3.3 Bayesian a posteriori formulation

This formulation, in terms of probability and likelihood, is the one used for the practical implementation of the FBA. Let us define the following probabilities:

$$\xi_t(i, j) = \Pr[X_t = i, X_{t+1} = j/Y, \lambda] \quad (3.27)$$

$$\gamma_t(i) = \Pr[X_t = i/Y, \lambda]. \quad (3.28)$$

Then the reestimation formulas become:

$$\begin{cases} \bar{a}_i &= \gamma_1(i) \\ \bar{a}_{ij} &= \sum_{t=1}^{L-1} \xi_t(i, j) / \sum_{t=1}^{L-1} \gamma_t(i) \\ \bar{b}_{ik} &= \sum_{\substack{t=1 \\ Y_t=k}}^L \gamma_t(i) / \sum_{t=1}^L \gamma_t(i), \end{cases} \quad (3.29)$$

where:

- $\sum_{t=1}^{L-1} \gamma_t(i)$ is the expected number of transitions out of state i .
- $\sum_{t=1}^{L-1} \xi_t(i, j)$ is the expected number of transitions from state i to state j .

- $\sum_{t=1}^L \delta(Y_t, k) \gamma_t(i)$ is the expected number of transitions out of state i , state i producing observation k .

Defining the probabilities:

$$\epsilon_t(i, j) = \Pr[X_t = i, X_{t+1} = j, Y/\lambda] \quad (3.30)$$

$$\chi_t(i) = \Pr[X_t = i, Y/\lambda], \quad (3.31)$$

and using Bayes' rule:

$$\epsilon_t(i, j) = \xi_t(i, j) \Pr(Y/\lambda) \quad (3.32)$$

$$\chi_t(i) = \gamma_t(i) \Pr(Y/\lambda). \quad (3.33)$$

The reestimation formulas become:

$$\left\{ \begin{array}{l} \bar{a}_i = \chi_1(i)/P \\ \bar{a}_{ij} = \sum_{t=1}^{L-1} \epsilon_t(i, j) / \sum_{t=1}^{L-1} \chi_t(i) \\ \bar{b}_{ik} = \sum_{\substack{t=1 \\ Y_t=k}}^L \chi_t(i) / \sum_{t=1}^L \chi_t(i). \end{array} \right. \quad (3.34)$$

Noting that:

$$\gamma_t(i) = \sum_{j=1}^J \xi_t(i, j) \quad (3.35)$$

$$\chi_t(i) = \sum_{j=1}^J \epsilon_t(i, j), \quad (3.36)$$

an alternate expression for the reestimation formulas is:

$$\begin{cases} \bar{a}_i &= \chi_1(i)/P \\ \bar{a}_{ij} &= \sum_{t=1}^{L-1} \epsilon_t(i, j) / \sum_{t=1}^{L-1} \chi_t(i) \\ \bar{b}_{ik} &= \sum_{t=1}^L \sum_{j=1}^J \epsilon_t(i, j) / \sum_{t=1}^L \chi_t(i). \end{cases} \quad (3.37)$$

Expressions (3.34) and (3.37) will be the most appropriate ones for practical implementation of the algorithm. Computations can be saved by expressing (3.37) in the following form:

$$\begin{cases} \bar{a}_i &= \chi_1(i)/P \\ \bar{a}_{ij} &= Na(i, j)/Da(i) \\ Na(i, j) &= \sum_{t=1}^{L-1} \epsilon_t(i, j) \\ Da(i) &= \sum_{j=1}^J Na(i, j) \\ \bar{b}_{ik} &= Nb(i, k)/Db(i) \\ Nb(i, k) &= \sum_{j=1}^J \sum_{t=1}^{L-1} \delta(Y_t, k) \epsilon_t(i, j) + \delta(Y_L, k) \chi_L(i) \\ Db(i) &= Da(i) + \chi_L(i) \end{cases} \quad (3.38)$$

($\delta(Y_t, k)$ is the Kronecker symbol).

3.2.4 Recursive evaluation of the growth transformation

The reestimation formulas (3.38) can be evaluated recursively in terms of the probability of partial sequences called the *forward* and *backward probabilities*. Let us define the joint probability $\alpha_t(i)$ of the partial observation sequence $Y(1:t)$ and state i , given the model λ :

$$\alpha_t(i) = \Pr[Y(1:t), X_t = i / \lambda], \quad (3.39)$$

and the conditional probability $\beta_t(i)$ of the partial observation sequence $Y(t+1:L)$ given state i and the model λ :

$$\beta_t(i) = \Pr[Y(t+1:L)/X_t = i, \lambda]. \quad (3.40)$$

(Most of the time the reference to the given model λ will be dropped for simplicity.) It is easily seen that these two probabilities can be computed recursively through:

- a forward recursion for $\alpha_t(i)$ (i.e., $\alpha_t(i)$ is computed as a function of $\alpha_{t-1}(i)$ as t increases):

$$\forall j = 1, s \quad \forall t \in [2, L] \quad \alpha_t(j) = \sum_{i=1}^s \alpha_{t-1}(i) a_{ij} b_j(Y_t) \quad (3.41)$$

- a backward recursion for $\beta_t(i)$ (i.e., $\beta_t(i)$ is computed as a function of $\beta_{t+1}(i)$ as t decreases):

$$\forall i = 1, s \quad \forall t \in [L-1, 1] \quad \beta_t(i) = \sum_{j=1}^s \beta_{t+1}(j) a_{ij} b_j(Y_{t+1}) \quad (3.42)$$

Moreover, the probabilities need to be initialized:

$$\begin{aligned} t = 1 \quad \text{for } \alpha_t(i): \quad \alpha_1(i) &= \Pr[Y(1), X_1 = i] \\ &= \Pr(X_1 = i) \Pr[Y(1)/X_1 = i] \\ t = L \quad \text{for } \beta_t(i): \quad \beta_{L-1}(i) &= \Pr[Y(L)/X_{L-1} = i] \\ &= \sum_{j=1}^s a_{ij} b_j(Y_L). \end{aligned}$$

Therefore:

$$\forall i = 1, s \quad \alpha_1(i) = \pi_1(i) b_i(Y_1). \quad (3.43)$$

The recursion (3.42) will hold for $t = L - 1$ if we initialize:

$$\forall j = 1, s \quad \beta_L(j) = 1. \quad (3.44)$$

Now it is possible to evaluate the probabilities $\epsilon_t(i, j)$ and $\chi_t(i)$ of (3.30) and (3.31) in terms of the forward and backward probabilities:

$$\forall i, j = 1, s \quad \forall t = 1, L \quad \begin{cases} \epsilon_t(i, j) = \alpha_t(i) a_{ij} b_j(Y_{t+1}) \beta_{t+1}(j) \\ \chi_t(i) = \alpha_t(i) \beta_t(i) \end{cases} \quad (3.45)$$

(note that: $\chi_L(i) = \alpha_L(i)$.)

Finally the likelihood P can be computed from:

$$\forall t \in [1, L] \quad P = \sum_{i=1}^s \alpha_t(i) \beta_t(i), \quad (3.46)$$

in particular for $t = L$:

$$P = \sum_{i=1}^s \alpha_L(i). \quad (3.47)$$

Combining (3.38), (3.45), and (3.47) we obtain the basic FBA of figure 3.1.

3.2.5 Computational complexity of the basic FBA

The computational complexity of the basic FBA is summarized in tables 3.1 and 3.3 for the number of operations and the CPU time. The computer speeds of table 3.3 should be compared with some of those available in table 3.2. For the models with which we are concerned, the complexity of the basic algorithm is

$$\begin{aligned}
& \forall i, j = 1, s \quad \forall k = 1, M \quad \text{evaluate:} \\
& \alpha_1(i) = a_i b_i(Y_1) \\
& t = 2, L \quad \alpha_t(j) = \sum_{i=1}^s \alpha_{t-1}(i) a_{ij} b_j(Y_t) \\
& \beta_L(i) = 1 \\
& t = L-1, 1 \quad \beta_t(i) = \sum_{j=1}^s \beta_{t+1}(j) a_{ij} b_j(Y_{t+1}) \\
& P = \sum_{i=1}^s \alpha_L(i) \\
& t = 1, L-1 \quad \epsilon_t(i, j) = \alpha_t(i) a_{ij} b_j(Y_{t+1}) \beta_{t+1}(j) \\
& Na(i, j) = \sum_{t=1}^{L-1} \epsilon_t(i, j) \\
& Da(i) = \sum_{j=1}^s Na(i, j) \\
& Nb(i, k) = \sum_{t=1}^{L-1} \sum_{j=1}^s \delta(Y_t, k) \epsilon_t(i, j) + \delta(Y_L, k) \alpha_L(i) \\
& Db(i) = Da(i) + \alpha_L(i) \\
& a_i = \alpha_1(i) \beta_1(i) / P \\
& a_{ij} = Na(i, j) / Da(i) \\
& b_{ik} = Nb(i, k) / Db(i)
\end{aligned}$$

Figure 3.1: Basic FBA (without any scaling).

roughly 3000 million floating point operations per iteration. For large models the number of operations per iteration is of the order $11s^2L$. The CPU times presented in table 3.3, however, are optimistic: in particular for large models, since the algorithm needs to access a large amount of data, and memory access time will slow down the algorithm quite significantly. As will be seen in the following sections, the basic FBA can be optimized (the number of operations can be decreased). To be practical, however, a scaling procedure must be performed on top of the algorithm, which in turn will increase the number of operations. Our effort will be to maintain this increase to a minimum.

Table 3.1: Number of operations for the basic FBA.

variable	No. of additions	No. of mult./div.	total
$\alpha_1(i)$	0	s	s
$\alpha_t(i)$	$s(s-1)(L-1)$	$2s^2(L-1)$	$3s^2(L-1) - s(L-1)$
$\beta_L(i)$	0	0	0
$\beta_t(i)$	$s(s-1)(L-1)$	$2s^2(L-1)$	$3s^2(L-1) - s(L-1)$
P	$s-1$	0	$s-1$
$\epsilon_t(i, j)$	0	$3s^2(L-1)$	$3s^2(L-1)$
$Na(i, j)$	$s^2(L-2)$	0	$s^2(L-2)$
$Da(i)$	$s(s-1)$	0	$s(s-1)$
$Nb(i, k)$	$s[1 + (s-1)(L-2)]$	0	$s^2(L-2) + s(3-L)$
$Db(i)$	s	0	s
a_i	0	$2s$	$2s$
a_{ij}	0	s^2	s^2
b_{ik}	0	sM	sM
total	$2s^2(2L-3)$ $+3s(2-L)-1$	$s^2(7L-6)$ $+s(M+3)$	$s^2(11L-12)+$ $s(M+9-3L)-1$
order of†	$4s^2L$	$7s^2L$	$11s^2L$

†for L large and $M \ll L$

Table 3.2: Some peak speeds of commercial computers.

Computer	Corporation	Peak Speed†
IBM-PC	IBM Co.	0.1
VAX-11/788	Digital Equipment Co.	1
MV/10000	Data General Co.	3
Cyber 855	Control Data Co.	12.5
Cyber 990	Control Data Co.	32.3
Cray 1	Cray Co.	160
Cray XMP-4	Cray Co.	1000

†in MFLOPS (Mega Floating Point Operations per Second)

except for the Cybers in MIPS (Mega Instructions Per Seconds)

Table 3.3: CPU time per iteration of the basic FBA.

model type	Number of operations	CPU time				
		0.1 MF	1 MF	10 MF	100 MF	1000 MF†
small ¹	3781	0.038s	0.004s	0.378ms	38 μ s	3.8 μ s
simple ²	260,064	2.601s	0.260s	0.026s	0.003s	0.260ms
large ³	2,691,856,959	7h28m39s	44m52s	4m29s	27s	2.7s

1: $s = 2, M = 6, L = 100$

2: $s = 5, M = 64, L = 1000$

3: $s = 64, M = 1024, L = 60000$

h: hours, m: minutes, s: seconds, ms: milli-seconds, μ s: micro-seconds

†MF=MFLOPS=Mega Floating Point Operations per Second

3.3 Practical implementation of the FBA

3.3.1 Introduction

The basic algorithm described above is efficient because it reduces the amount of computations from a factor exponential in L (of the order of Ls^L) for a direct maximization technique, to a factor linear in L (of the order of s^2L) for the FBA. Unfortunately the algorithm is still not ready for practical implementation because:

- some of the variables become very small (both as t or the number of iterations increases), resulting in underflow problems.
- undesirable zero entries can be generated in the matrices π_1 , A , B in one of the following ways:
 - a zero entry in one of the initial estimates will be reestimated to 0 (as previously seen in section 3.2.3.1).
 - an entry not represented in the finite training data set will be estimated and reestimated to 0.

These zero entries are undesirable because:

- they accentuate the underflow problems,
- they prevent the use of such functions as logarithm without checking the arguments,
- they provide wrong estimations of small non-zero parameters not represented in the finite training data set.

The underflow problem can be solved by introducing a *scaling procedure* on top of the FBA. The zero entries problem will be solved by constraining the model parameters λ to be no less than a minimum non-zero lower bound ².

²An $N \times M$ stochastic matrix D satisfying the constraints (3.11) needs to be renormalized when its "zero" entries, i.e., entries less than a minimum m , are set to this minimum m . The

3.3.2 Scaling the forward and backward probabilities

Basically $\alpha_t(i)$ (resp. $\beta_t(i)$) is the probability of a sequence of length t (resp. $L-t$). This probability is the product of individual probabilities for each member in the sequence. Probabilities being less than 1, $\alpha_t(i) \rightarrow 0$ as the sequence length $t \rightarrow \infty$. Let us consider the "equiprobable case:"

$$\forall i, j = 1, s \quad \forall k = 1, M \quad a_i = 1/s, \quad a_{ij} = 1/s, \quad b_{ik} = 1/M.$$

Then from the α -recursion (3.41) or the definition (3.39), it is easily seen that:

$$\forall j = 1, s \quad \forall t \geq 1 \quad \alpha_t(j) = \frac{1}{sM^t},$$

which shows that $\alpha_t(j)$ exponentially tends to 0 as $t \rightarrow \infty$, resulting in fast occurring underflow problems. The case of $\beta_t(i)$ is quite similar to that of $\alpha_t(i)$, with a time axis reversal. Therefore:

$$\forall i = 1, s \quad \begin{cases} \alpha_t(i) \rightarrow 0 & \text{when } t \rightarrow \infty \\ \beta_t(i) \rightarrow 0 & \text{when } t \rightarrow 1 \text{ and } L \rightarrow \infty. \end{cases}$$

The scaling procedure to be defined should constrain $\alpha_t(i)$ and $\beta_t(i)$ to be no less than a strictly positive, constant lower bound. From the recursions (3.41) and (3.42) it appears that one needs to constrain the model parameters to the following:

$$\forall i, j = 1, s \quad \forall k = 1, M \quad \begin{cases} 0 < m_i \leq a_i \leq 1 \\ 0 < m_a \leq a_{ij} \leq 1 \\ 0 < m_b \leq b_{ik} \leq 1. \end{cases} \quad (3.48)$$

renormalization is not necessary if $m \ll 1$. If J_i is the set of indices of the zero entries in row i , of cardinality N_i , renormalise to:

$$\forall j \in J_i \quad d_{ij}^* = m$$

$$\forall j \notin J_i \quad d_{ij}^* = (1 - N_i m) d_{ij} / \sum_{k \notin J_i}^M d_{ik}$$

From the α -recursion (3.41) it follows that:

$$\forall j = 1, s \quad \forall t \geq 1 \quad m_a m_b \underbrace{\sum_{i=1}^s \alpha_{t-1}(i)}_0 \leq \alpha_t(j) \leq \underbrace{\sum_{i=1}^s \alpha_{t-1}(i)}_0.$$

To constrain $\alpha_t(j)$ to a fixed strictly positive range, we define the scaled variable:

$$\forall j = 1, s \quad \forall t \geq 1 \quad \alpha'_t(j) = \frac{\alpha_t(j)}{\sum_{i=1}^s \alpha_{t-1}(i)},$$

which leads to the desired range:

$$\forall j = 1, s \quad \forall t \geq 1 \quad 0 < m_a m_b \leq \alpha'_t(j) \leq 1.$$

The α -scaling procedure, combined with (3.43) and (3.41), is more precisely defined below, in terms of the *local scaling variable* for $\alpha_t(i)$, c_t :

• initialization:

$$\left\{ \begin{array}{ll} \forall i = 1, s & \alpha_1^*(i) = a_i b_i(Y_1) \\ & c_1 = \sum_{i=1}^s \alpha_1^*(i) \\ \forall i = 1, s & \alpha'_1(i) = \alpha_1^*(i)/c_1 \end{array} \right. \quad (3.49)$$

• recursion:

$$\forall t = 2, L \quad \left\{ \begin{array}{ll} \forall j = 1, s & \alpha_t^*(j) = \sum_{i=1}^s \alpha'_{t-1}(i) a_{ij} b_j(Y_t) \\ & c_t = \sum_{i=1}^s \alpha_t^*(i) \\ \forall j = 1, s & \alpha'_t(j) = \alpha_t^*(j)/c_t \end{array} \right. \quad (3.50)$$

$\alpha_t^*(i)$ is an auxiliary variable, not equal to $\alpha_t(i)$ except for $t = 1$. Noting that:

$$\sum_{j=1}^s \alpha'_t(j) = \left(\sum_{j=1}^s \alpha_t^*(j) \right) / c_t \quad \text{i.e.,}$$

$$\forall t \geq 1 \quad \sum_{j=1}^s \alpha'_t(j) = 1, \quad (3.51)$$

the ranges of the α -variables are:

$$\begin{aligned} 0 < m_a m_b &\leq \alpha_t^*(j) \leq 1 \\ 0 < m_a m_b / s &\leq \alpha_t'(j) \leq 1 \\ 0 < s m_a m_b &\leq c_t \leq s, \end{aligned} \quad (3.52)$$

which solves the underflow problem for $\alpha_t(j)$.

Now $\alpha_t(i)$, $\alpha_t^*(i)$, and $\alpha_t'(i)$ can be directly related to one another through a *global scaling variable* U_t :

$$\forall t \geq 1 \quad \begin{cases} U_t = c_t U_{t-1} \\ \alpha_t = U_t \alpha_t' \\ \alpha_t = U_{t-1} \alpha_t^* \end{cases} \quad (3.53)$$

with:

$$U_0 = 1 \quad \text{and} \quad U_t = \prod_{n=1}^t c_n. \quad (3.54)$$

Or in terms of log-variables (for any variable a we define $\hat{a} = \log_{10} a$):

$$\forall t \geq 1 \quad \begin{cases} \hat{U}_t = \hat{c}_t + \hat{U}_{t-1} \\ \hat{\alpha}_t = \hat{U}_t + \hat{\alpha}_t' \\ \hat{\alpha}_t = \hat{U}_{t-1} + \hat{\alpha}_t^* \end{cases} \quad (3.55)$$

with:

$$\hat{U}_0 = 0 \quad \text{and} \quad \hat{U}_t = \sum_{n=1}^t \hat{c}_n. \quad (3.56)$$

The proof of (3.53) is easily derived by induction:

- equation (3.54) is true for $t = 1$: $U_1 = c_1$, $\alpha_1 = c_1 \alpha_1'$, $\alpha_1 = \alpha_1^*$.
- let us assume (3.53) is true for t and let us prove it is true for $t + 1$:

$$\begin{aligned}
\forall j = 1, s \quad \alpha_{t+1}(j) &= \sum_{i=1}^s \alpha_t(i) a_{ij} b_j(Y_{t+1}) \\
&= U_t \sum_{i=1}^s \alpha'_t(i) a_{ij} b_j(Y_{t+1}) = U_t \alpha_{t+1}^*(j) \\
&= U_t c_{t+1} \alpha'_{t+1}(j) = U_{t+1} \alpha'_{t+1}(j),
\end{aligned}$$

therefore,

$$\begin{cases} U_{t+1} &= c_{t+1} U_t \\ \alpha_{t+1}(j) &= U_{t+1} \alpha'_{t+1}(j) \\ \alpha_{t+1}(j) &= U_t \alpha_{t+1}^*(j) \quad \text{q.e.d} \end{cases}$$

(3.54) is derived from a straightforward induction of $U_0 = 1$ and $U_t = c_t U_{t-1}$.

The β -scaling procedure is similar, and respectively defines a local and global scaling variable s_t and V_t . It is summarized below:

• initialization:

$$\begin{cases} \forall i = 1, s \quad \beta_L^*(i) &= 1 \\ &s_L = \sum_{i=1}^s \beta_L^*(i) = s \\ \forall i = 1, s \quad \beta_L'(i) &= \beta_L^*(i)/s = 1/s \end{cases} \quad (3.57)$$

• recursion:

$$\forall t = L-1, 1 \begin{cases} \forall i = 1, s \quad \beta_t^*(i) &= \sum_{j=1}^s \beta_{t+1}'(j) a_{ij} b_j(Y_{t+1}) \\ &s_t = \sum_{j=1}^s \beta_t^*(j) \\ \forall j = 1, s \quad \beta_t'(i) &= \beta_t^*(i)/s_t \end{cases} \quad (3.58)$$

• the ranges of the β -variables are:

$$\begin{aligned}
0 < m_a m_b &\leq \beta_t^*(i) \leq 1 \\
0 < m_a m_b / s &\leq \beta_t'(i) \leq 1 \\
0 < s m_a m_b &\leq s_t \leq s
\end{aligned} \tag{3.59}$$

• global scaling:

$$\forall t = L, 1 \quad \left\{ \begin{array}{l} V_t = s_t V_{t+1} \\ \beta_t = V_t \beta_t' \\ \beta_t = V_{t+1} \beta_t^* \end{array} \right. \tag{3.60}$$

with:

$$V_{L+1} = 1 \quad \text{and} \quad V_t = \prod_{n=t}^L s_n. \tag{3.61}$$

Or in terms of log-variables:

$$\forall t = L, 1 \quad \left\{ \begin{array}{l} \hat{V}_t = \hat{s}_t + \hat{V}_{t+1} \\ \hat{\beta}_t = \hat{V}_t + \hat{\beta}_t' \\ \hat{\beta}_t = \hat{V}_{t+1} + \hat{\beta}_t^* \end{array} \right. \tag{3.62}$$

with:

$$\hat{V}_{L+1} = 0 \quad \text{and} \quad \hat{V}_t = \sum_{n=t}^L \hat{s}_n. \tag{3.63}$$

Note that:

$$s_L = V_L = s, \tag{3.64}$$

and that the likelihood can be expressed as follows:

$$\begin{aligned}
P &= \sum_{i=1}^s \alpha_L(i) \\
&= U_L \underbrace{\sum_{i=1}^s \alpha_L'(i)}_1 \quad \text{i.e.,}
\end{aligned} \tag{3.65}$$

$$P = U_L, \quad (3.66)$$

and from (3.45) and (3.53):

$$\chi_L(i) = \alpha_L(i) \quad (3.67)$$

$$\chi_L(i) = U_L \alpha'_L(i). \quad (3.68)$$

3.3.3 Reestimation in terms of the scaled variables

The reestimation formulas (3.38) are expressed in terms of $\epsilon_t(i, j)$ evaluated in (3.45). Using only the scaled variables, we have:

$$\forall t = 1, L-1 \quad \epsilon_t(i, j) = U_t \alpha'_t(i) a_{ij} b_j(Y_{t+1}) V_{t+1} \beta'_{t+1}(j).$$

Defining:

$$W_t = U_t V_{t+1}, \quad (3.69)$$

we obtain:

$$\epsilon_t(i, j) = W_t \alpha'_t(i) a_{ij} b_j(Y_{t+1}) \beta'_{t+1}(j). \quad (3.70)$$

Two algorithms based on two different scaled versions of $\epsilon_t(i, j)$ will be presented below. The first version will be referred to as the "FBA with partial scaling." It is a special case ($s_t = c_t$) of the second algorithm called the "FBA with full scaling" ($s_t \neq c_t$). They operate a different compromise between algorithm accuracy and algorithm speed. The first algorithm should be used when greater speed is required at the expense of less accuracy (the full dynamic range of the computer is not used, however the accuracy of the algorithm will be sufficient for most practical speech applications). The second algorithm should be used when the maximum available accuracy is needed (the full dynamic range of the computer is used) and a slower algorithm is affordable (small models, a short training sequence, or a very fast computer).

3.3.3.1 A partial scaling procedure: $s_t = c_t$

This scaling procedure for which $s_t = c_t$ (c_t being defined in the preceding section by (3.49) and (3.50)) has the advantage of leaving the reestimation formulas invariant under the scaling transformation. If in the original unscaled reestimation formulas $\epsilon_t(i, j)$ is replaced by:

$$\epsilon_t^*(i, j) = \alpha'_t(i) a_{ij} b_j(Y_{t+1}) \beta'_{t+1}(j), \quad (3.71)$$

then the reestimated parameters λ remain unchanged. This is easily seen from the fact that for $s_t = c_t$, W_t is a constant independent of t :

$$W_t = \prod_{n=1}^L c_n = W_1 = U_L = V_1.$$

Therefore in both the numerators and denominators of the reestimation formulas, W_t can be factored out of the sums and cancelled out.

For the transition matrix A from (3.38):

$$\begin{aligned} a_{ij} &= \frac{\sum_{t=1}^{L-1} \epsilon_t(i, j)}{\sum_{j=1}^s \sum_{t=1}^{L-1} \epsilon_t(i, j)} \\ &= \frac{\sum_{t=1}^{L-1} W_t \alpha'_t(i) a_{ij} b_j(Y_{t+1}) \beta'_{t+1}(j)}{\sum_{j=1}^s \sum_{t=1}^{L-1} W_t \alpha'_t(i) a_{ij} b_j(Y_{t+1}) \beta'_{t+1}(j)} \\ a_{ij} &= \frac{\sum_{t=1}^{L-1} \epsilon_t^*(i, j)}{\sum_{j=1}^s \sum_{t=1}^{L-1} \epsilon_t^*(i, j)}. \end{aligned}$$

Similarly for the output probability matrix B , from (3.36) and (3.37):

$$\begin{aligned} b_{ik} &= \frac{\sum_{t=1}^L \sum_{j=1}^s \delta(Y_t, k) \epsilon_t(i, j)}{\sum_{t=1}^L \sum_{j=1}^s \epsilon_t(i, j)} \\ &= \frac{\sum_{t=1}^{L-1} \sum_{j=1}^s \delta(Y_t, k) \epsilon_t(i, j) + \delta(Y_L, k) \chi_L(i)}{\sum_{t=1}^{L-1} \sum_{j=1}^s \epsilon_t(i, j) + \chi_L(i)} \\ &= \frac{\sum_{t=1}^{L-1} \sum_{j=1}^s \delta(Y_t, k) W_t \epsilon_t^*(i, j) + \delta(Y_L, k) U_L \alpha'_L(i)}{\sum_{t=1}^{L-1} \sum_{j=1}^s W_t \epsilon_t^*(i, j) + U_L \alpha'_L(i)} \end{aligned}$$

$$b_{ik} = \frac{\sum_{t=1}^L \sum_{j=1}^s \delta(Y_t, k) \epsilon_i^*(i, j)}{\sum_{t=1}^L \sum_{j=1}^s \epsilon_i^*(i, j)}.$$

For the distribution of the states π_1 , from figure (3.1):

$$\begin{aligned} a_i &= \alpha_1(i) \beta_1(i) / P \\ &= U_1 V_1 \alpha'_1(i) \beta'_1(i) / P \quad \text{with} \quad U_1 V_1 = c_1 U_L = c_1 P \\ a_i &= c_1 \alpha'_1(i) \beta'_1(i). \end{aligned}$$

Unfortunately this scaling procedure, valid for $\alpha_t(i)$, does not guarantee that $\beta_t(i)$ will not underflow. If the training sequence is symmetric from its middle point, then $\alpha_t(i)$ and $\beta_t(i)$ are symmetrically related and a scaling $s_t = K c_{L-t+1}$ would be appropriate. But in general scaling $\beta_t(i)$ by the same amount as $\alpha_t(i)$ would still lead to practical underflow or overflow. As seen from (3.57) and (3.58) with $s_t = c_t$, for $t = L - 1, 1$:

$$m_a m_b \frac{\sum_{j=1}^s \beta_{t+1}^*(i)}{c_{t+1}} \leq \beta_t^*(i) \leq \frac{\sum_{j=1}^s \beta_{t+1}^*(i)}{c_{t+1}}.$$

Here, contrary to the α -scaling, $\frac{\sum_{j=1}^s \beta_{t+1}^*(i)}{c_{t+1}} \neq 1$. Given that $\beta_L^*(i) = 1$ and from (3.52) $s m_a m_b \leq c_{t+1} \leq s$ we obtain:

$$\begin{aligned} m_a m_b &\leq \beta_{L-1}^*(i) \leq 1/m_a m_b \\ \underbrace{(m_a m_b)^t}_0 &\leq \beta_{L-t}^*(i) \leq \underbrace{1/(m_a m_b)^t}_\infty. \end{aligned}$$

The proof of the existence of constant (independent of t) non-zero bounds for $\beta_t^*(i)$ based on $s_t = c_t$ is not known to us; therefore special precautions need to be taken when implementing the algorithm. The FBA with partial scaling is summarized in figure 3.2. The largest permissible floating point number (computer range) is denoted 10^R .

$\forall i, j = 1, s \quad \forall k = 1, M \quad \text{evaluate:}$

$\alpha'_t(i), c_t, \beta'_t(i) \quad \text{for } t = 1, L^\dagger$

$$\mathcal{L} = \sum_{t=1}^L \hat{c}_t$$

$$t = 1, L - 1 \quad \hat{\epsilon}_t^*(i, j) = \hat{\alpha}'_t(i) + \hat{a}_{ij} + \hat{b}_{ij}(Y_{t+1}) + \hat{\beta}'_{t+1}(j)$$

$$Na(i, j) = \sum_{\substack{t=1 \\ |i_t^*| < R}}^{L-1} \epsilon_t^*(i, j)$$

$$Da(i) = \sum_{j=1}^s Na(i, j)$$

$$Db(i) = Da(i) + \alpha'_L(i)$$

$$Nb(i, k) = \sum_{\substack{t=1 \\ |i_t^*| < R}}^{L-1} \sum_{j=1}^s \delta(Y_t, k) \epsilon_t^*(i, j) + \delta(Y_L, k) \alpha'_L(i)$$

$$a_{ij} = Na(i, j) / Da(i)$$

$$b_{ik} = Nb(i, k) / Db(i)$$

$$a_i = c_1 \alpha'_1(i) \beta'_1(i)$$

\dagger see (3.49), (3.50), (3.57), (3.58) with $s_t = c_t$

Figure 3.2: FBA with partial scaling.

To optimize the algorithm further one might take advantage of the fact that the products $a_i b_{jk}$ appear 3 times in the algorithm: for the evaluation of $\alpha_t(i)$, $\beta_t(i)$, and $\epsilon_t(i, j)$. If the memory capacity permits it, these products can be computed only once and stored in an array. Similarly, products of the form $a_i b_j (Y_{t+1}) \beta_{t+1}(j)$ appear in the evaluation of both $\beta_t(j)$ and $\epsilon_t(i, j)$. Such products can be computed only once if $\beta_t(i)$, $Na(i, j)$, and $Nb(i, k)$ are evaluated "simultaneously" over a backward loop for $t = L - 1, 1$. For this purpose let us define:

$$\forall j = 1, s \quad \forall t = L - 1, 1 \quad u(j, t) = b_j(Y_{t+1}) \beta'_{t+1}(j). \quad (3.72)$$

The algorithm is reformulated in figure 3.3. Note that the sums in $\beta'_t(i)$, $Na(i, j)$, and $Nb(i, k)$ are meant over the significant terms, i.e., the terms whose magnitudes are less than the computer range 10^R .

If more storage is available this algorithm can still be improved by noticing that:

$$\begin{aligned} \sum_{k=1}^M \sum_{t=1}^{L-1} \delta(Y_t, k) \alpha'_t(i) u(j, t) &= \sum_{t=1}^{L-1} \sum_{k=1}^M \delta(Y_t, k) \alpha'_t(i) u(j, t) \\ &= \sum_{t=1}^{L-1} \underbrace{\left[\sum_{k=1}^M \delta(Y_t, k) \right]}_1 \alpha'_t(i) u(j, t) \\ &= \sum_{t=1}^{L-1} \alpha'_t(i) u(j, t). \end{aligned}$$

This version of the algorithm is shown in figure 3.4.

$\forall i, j = 1, s \quad \forall k = 1, M$ evaluate:

$$\alpha'_t(i), c_t \quad \text{for } t = 1, L^\dagger$$

$$\mathcal{L} = \sum_{t=1}^L \hat{c}_t$$

$$\beta'_L(i) = 1/c_L$$

$$t = L - 1, 1 \quad \hat{u}(j, t) = \hat{b}_j(Y_{t+1}) + \hat{\beta}'_{t+1}(j)$$

$$t = L - 1, 1 \quad \beta'_t(i) = \frac{1}{c_t} \sum_{j=1}^s a_{ij} u(j, t)$$

$$Na(i, j) = a_{ij} \sum_{t=1}^{L-1} \alpha'_t(i) u(j, t)$$

$$Da(i) = \sum_{j=1}^s Na(i, j)$$

$$Db(i) = Da(i) + \alpha'_L(i)$$

$$Nb(i, k) = \sum_{j=1}^s a_{ij} \sum_{t=1}^{L-1} \delta(Y_t, k) \alpha'_t(i) u(j, t) + \delta(Y_L, k) \alpha'_L(i)$$

$$a_{ij} = Na(i, j) / Da(i)$$

$$b_{ik} = Nb(i, k) / Db(i)$$

$$a_i = c_1 \alpha'_1(i) \beta'_1(i)$$

† see (3.49), (3.50)

Figure 3.3: Optimized FBA with partial scaling.

$\forall i, j = 1, s \quad \forall k = 1, M \quad \text{evaluate:}$

$$\alpha'_t(i), c_t \quad \text{for } t = 1, L^\dagger$$

$$\mathcal{L} = \sum_{t=1}^L \hat{c}_t$$

$$\beta'_L(i) = 1/c_L$$

$$t = L - 1, 1 \quad \hat{u}(j, t) = \hat{b}_j(Y_{t+1}) + \hat{\beta}'_{t+1}(j)$$

$$t = L - 1, 1 \quad \beta'_t(i) = \frac{1}{c_t} \sum_{j=1}^s a_{ij} u(j, t)$$

$$Nb(i, j, k) = \sum_{t=1}^{L-1} \delta(Y_t, k) \alpha'_t(i) u(j, t)$$

$$Na(i, j) = a_{ij} \sum_{k=1}^M Nb(i, j, k)$$

$$Da(i) = \sum_{j=1}^s Na(i, j)$$

$$Db(i) = Da(i) + \alpha'_L(i)$$

$$Nb(i, k) = \sum_{j=1}^s a_{ij} Nb(i, j, k) + \delta(Y_L, k) \alpha'_L(i)$$

$$a_{ij} = Na(i, j) / Da(i)$$

$$b_{ik} = Nb(i, k) / Db(i)$$

$$a_i = c_1 \alpha'_1(i) \beta'_1(i)$$

\dagger see (3.49), (3.50)

Figure 3.4: Further-optimized FBA with partial scaling.

At that stage one might be quite satisfied with the above optimized versions of the partially scaled FBA. However one feature of the algorithm is still of concern: the use of logarithms in some of the variables to prevent underflow. The problem with logarithms is that they are computationally very expensive (one logarithm is roughly equivalent to 20 multiplications). Can we implement a logarithm-free algorithm which would preserve the accuracy of the algorithms developed so far? It turns out that it will be possible to do so if we take advantage of the inequalities (3.48) and (3.52) derived in section 3.3.2. After the scaling, most of the remaining underflow problems come from evaluating $\epsilon_i^*(i, j)$ of (3.71) as a product of very small quantities. Hopefully the ranges of three of the variables have been bounded:

$$\begin{aligned} m_a &\leq a_{ij} && \leq 1 \\ m_b &\leq b_j(Y_{i+1}) && \leq 1 \\ m_a m_b / s &\leq \alpha'_i(i) && \leq 1. \end{aligned}$$

Similarly we will constrain the range of $\beta'_{i+1}(j)$ to:

$$0 < \beta_R \leq \beta'_{i+1}(j), \quad (3.73)$$

so that the overall range of $\epsilon_i^*(i, j)$ stays in the dynamic range of the computer. The β'_{i+1} variables outside that range will be discarded, i.e., the sums of the form $\sum_i \epsilon_i^*(i, j)$ will be truncated only to the significant terms. What value of β_R would be appropriate?

Let $m_a = m_b = m = 10^{-\nu}$ ($\nu > 0$), then:

$$\begin{aligned} 10^{-\nu} &\leq a_{ij} && \leq 1 \\ 10^{-\nu} &\leq b_j(Y_{i+1}) && \leq 1 \\ 10^{-2\nu}/s &\leq \alpha'_i(i) && \leq 1 \\ \beta_R 10^{-4\nu}/s &\leq \epsilon_i^*(i, j) && \leq 1. \end{aligned}$$

How should the matrix entries be constrained, i.e., what ν should one choose? The β variables play a role analogous to that of the α variables. Therefore, since we would like their range to be constrained to the same range as the α 's, i.e., $\beta_R = 10^{-2\nu}/s$, we have:

$$10^{-6\nu}/s^2 \leq \epsilon_i^*(i, j).$$

Choosing $10^{-6\nu}/s^2 = 10^{-R}$ will solve the problem, i.e.:

$$\nu = \frac{1}{6}(R - 2\hat{s}). \quad (3.74)$$

In practice $R = 74$ (computer range 10^R), and $s = 64$ lead to $\nu = 11$ and $\beta_R = 2 \times 10^{-24}$. This algorithm is faster than the previous algorithm (by a little more than a factor 20) and is quite accurate. As a matter of fact the only numerical differences come from the very small model entries around $m = 10^{-\nu}$ which represent 0.

3.3.3.2 The general scaling procedure: $s_i \neq c_i$

This scaling procedure guarantees that $\alpha'_i(i)$ and $\beta'_i(i)$ stay in the dynamic range of the computer (as seen above). However the reestimation formulas *do not* stay invariant under the general scaling procedure but can be expressed in terms of the scaled variables $\epsilon'_i(i, j)$ defined below and a global correcting multiplying factor (see expressions (3.83), (3.85), and (3.78) below). Naturally $\epsilon_i(i, j)$ in expression (3.70) is outside the dynamic range of the computer and $\sum_i \epsilon_i(i, j)$ cannot be evaluated directly. However we can keep track of the log-variables:

$$\hat{\epsilon}_i(i, j) = \hat{W}_i + \hat{\alpha}'_i(i) + \hat{a}_{ij} + \hat{b}_j(Y_{i+1}) + \hat{\beta}'_{i+1}(j) \quad (3.75)$$

$$\hat{W}_i = \hat{U}_i + \hat{V}_{i+1}. \quad (3.76)$$

Another useful recursion for W_i involves only one global scaling variable:

$$\hat{W}_i = \hat{W}_{i+1} + \hat{U}_i - \hat{U}_{i+1} + \hat{s}_{i+1}. \quad (3.77)$$

This is easily derived from (3.62) and (3.76):

$$\begin{aligned}\hat{W}_t &= \hat{U}_t + \hat{V}_{t+1} \\ \hat{V}_{t+1} &= \hat{s}_{t+1} + \hat{V}_{t+2} \\ \hat{W}_{t+1} &= \hat{U}_{t+1} + \hat{V}_{t+2}.\end{aligned}$$

Let $[-10^R, 10^R]$ be the dynamic range of the computer³, and S a security margin ($0 < S < R$). The general scaling procedure will be the following: find the largest term in the sum, "translate" all the terms in the sum by an amount such that the largest term become 10^S , take the common scaling factor outside the sum, and perform the sum only on the significant scaled terms.

• reestimation for a_i :

$$\begin{aligned}a_i &= \frac{\chi_1(i)}{P} = \frac{\alpha_1(i)\beta_1(i)}{P} \\ a_i &= \frac{U_1 V_1}{P} \alpha'_1(i) \beta'_1(i)\end{aligned}\quad (3.78)$$

• reestimation for a_{ij} :

Let:

$$\hat{M}(i, j) = \max_{t=1, L-1} \hat{\epsilon}_t(i, j). \quad (3.79)$$

Translate $\epsilon_t(i, j)$ on the range $[-10^R, 10^R]$ where S is a security factor such that $\sum_t \epsilon'_t(i, j)$ do not exceed 10^R (avoid overflow):

$$\begin{aligned}\max\left[\sum_{t=1}^L \epsilon'_t(i, j)\right] &\leq 10^R \quad \text{i.e.,} \\ \max\left[\sum_{t=1}^L \epsilon'_t(i, j)\right] &\leq \sum_{t=1}^L \max \epsilon'_t(i, j) \leq \sum_{t=1}^L 10^S \leq L \times 10^S,\end{aligned}$$

therefore choose S such that $L \times 10^S = 10^R$ i.e.,

³More precisely the range is of the form $[-10^{R_1}, -10^{-R_2}] \cup \{0\} \cup [10^{-R_2}, 10^{R_1}]$. If $R_1 \neq R_2$ both can be incorporated into the algorithms where appropriate, or one can choose $R = \min(R_1, R_2)$.

$$S = R - \hat{L}. \quad (3.80)$$

Define $\epsilon'_t(i, j) = 10^{S - \hat{M}(i, j)} \epsilon_t(i, j)$:

$$\hat{\epsilon}'_t(i, j) = S - \hat{M}(i, j) + \hat{\epsilon}_t(i, j). \quad (3.81)$$

Redefine the reestimation formulas (3.38) in terms of $\hat{\epsilon}'_t(i, j)$:

$$a_{ij} = Na(i, j) / Da(i)$$

$$Na(i, j) = \sum_{t=1}^{L-1} \epsilon_t(i, j) = 10^{\hat{M}(i, j) - S} \sum_{t=1}^{L-1} \epsilon'_t(i, j).$$

Let $Na'(i, j) = \sum_{t=1}^{L-1} \epsilon'_t(i, j)$, then:

$$Na(i, j) = 10^{\hat{M}(i, j) - S} Na'(i, j)$$

$$Da(i) = \sum_{j=1}^s Na(i, j) = \sum_{j=1}^s 10^{\hat{M}(i, j) - S} Na'(i, j).$$

Let:

$$\mu(i) = \max_{j=1, s} [10^{\hat{M}(i, j) - S} Na'(i, j)],$$

and define a new security margin:

$$S' = R - \hat{s}. \quad (3.82)$$

Define:

$$Na''(i, j) = 10^{S' - \mu(i)} [10^{\hat{M}(i, j) - S} Na'(i, j)]$$

$$Da''(i) = \sum_{j=1}^s Na''(i, j).$$

Then:

$$Da(i) = \sum_{j=1}^s 10^{\hat{\mu}(i)-s'} Na''(i, j) = 10^{\hat{\mu}(i)-s'} Da''(i).$$

Therefore:

$$a_{ij} = 10^{\hat{M}(i,j)-\hat{\mu}(i)+s'-s} Na'(i, j) / Da''(i). \quad (3.83)$$

It is of the form:

$$a_{ij} = K_a(i, j) \frac{Na'(i, j)}{Da''(i)}. \quad (3.84)$$

• reestimation for b_{ik} :

$$b_{ik} = Nb(i, k) / Db(i)$$

$$Nb(i, k) = \sum_{j=1}^s Nb(i, j, k) + \delta(Y_L, k) \alpha_L(i)$$

with:

$$Nb(i, j, k) = \sum_{t=1}^{L-1} \delta(Y_t, k) \epsilon_t(i, j)$$

$$Nb(i, j, k) = 10^{\hat{M}(i,j)-s} \sum_{t=1}^{L-1} \delta(Y_t, k) \epsilon'_t(i, j).$$

Define $Nb'(i, j, k) = \sum_{t=1}^{L-1} \delta(Y_t, k) \epsilon'_t(i, j)$, then:

$$Nb(i, k) = \sum_{j=1}^s 10^{\hat{M}(i,j)-s} Nb'(i, j, k) + \delta(Y_L, k) \alpha_L(i).$$

Let:

$$\eta(i, k) = \max_{j=1, s} [10^{\hat{M}(i,j)-s} Nb'(i, j, k)],$$

and:

$$Nb''(i, j, k) = 10^{s'-\eta(i,k)} [10^{\hat{M}(i,j)-s} Nb'(i, j, k)],$$

then:

$$Nb(i, k) = 10^{\eta(i,k)-s'} \sum_{j=1}^s Nb''(i, j, k) + \delta(Y_L, k) U_L \alpha'_L(i).$$

Let:

$$Nb'(i, k) = \sum_{j=1}^s Nb''(i, j, k) + 10^{s'-\eta(i,k)} \delta(Y_L, k) U_L \alpha'_L(i),$$

then:

$$Nb(i, k) = 10^{\hat{\eta}(i, k) - S'} Nb'(i, k)$$

$$Db(i) = Da(i) + \alpha_L(i)$$

$$Db(i) = 10^{\hat{\mu}(i) - S'} \sum_{j=1}^{\bullet} Na''(i, j) + U_L \alpha'_L(i).$$

Let:

$$Db''(i) = \sum_{j=1}^{\bullet} Na''(i, j) + 10^{S' - \hat{\mu}(i)} U_L \alpha'_L(i),$$

then:

$$b_{ik} = 10^{\hat{\eta}(i, k) - \hat{\mu}(i)} Nb'(i, k) / Db''(i). \quad (3.85)$$

It is of the form:

$$b_{ik} = K_b(i, k) \frac{Nb'(i, k)}{Db''(i)}. \quad (3.86)$$

The FBA with full scaling is summarized in figure 3.5.

$\forall i, j = 1, s \quad \forall k = 1, M \quad \text{evaluate:}$

$$\begin{aligned}
 & \alpha'_t(i), \beta'_t(i), U_t, V_t, W_t \quad \text{for } t = 1, L^\dagger \\
 & t = 1, L-1 \quad \hat{\epsilon}_t(i, j) = \hat{W}_t + \hat{\alpha}'_t(i) + \hat{a}_{ij} + \hat{b}_j(Y_{t+1}) + \hat{\beta}'_{t+1}(j) \\
 & \quad \hat{M}(i, j) = \max_{t=1, L-1} \hat{\epsilon}_t(i, j) \\
 & t = 1, L-1 \quad \hat{\epsilon}'_t(i, j) = S - \hat{M}(i, j) + \hat{\epsilon}_t(i, j) \\
 & \quad Na'(i, j) = \sum_{\substack{t=1 \\ |\hat{\epsilon}'_t(i, j)| < R}}^{L-1} \hat{\epsilon}'_t(i, j) \\
 & Nb'(i, j, k) = \sum_{t=1}^{L-1} \delta(Y_t, k) \hat{\epsilon}'_t(i, j) \\
 & \quad \hat{\mu}(i) = \max_{j=1, s} [\hat{M}(i, j) - S + \hat{Na}'(i, j)] \\
 & \quad \hat{\eta}(i, k) = \max_{j=1, s} [\hat{M}(i, j) - S + \hat{Nb}'(i, j, k)] \\
 & \quad \hat{Na}''(i, j) = \hat{M}(i, j) - \hat{\mu}(i) + S' - S + \hat{Na}'(i, j) \\
 & \quad \hat{Nb}''(i, j, k) = \hat{M}(i, j) - \hat{\eta}(i, k) + S' - S + \hat{Nb}'(i, j, k) \\
 & \quad Da''(i) = \sum_{\substack{j=1 \\ |\hat{Na}''(i, j)| < R}}^s \hat{Na}''(i, j) \\
 & \quad Db''(i) = Da''(i) + 10^{S' - \hat{\mu}(i)} U_L \alpha'_L(i) \\
 & Nb'(i, k) = \sum_{\substack{j=1 \\ |\hat{Nb}''(i, j, k)| < R}}^s \hat{Nb}''(i, j, k) + 10^{S' - \hat{\eta}(i, k)} \delta(Y_L, k) U_L \alpha'_L(i) \\
 & \quad \hat{a}_{ij} = \hat{M}(i, j) - \hat{\mu}(i) + S' - S + \hat{Na}'(i, j) - \hat{Da}''(i) \\
 & \quad \hat{b}_{ik} = \hat{\eta}(i, k) - \hat{\mu}(i) + \hat{Nb}'(i, k) - \hat{Db}''(i) \\
 & \quad \hat{a}_i = \hat{U}_1 + \hat{V}_1 + \hat{\alpha}'_1(i) + \hat{\beta}'_1(i) - \hat{P}
 \end{aligned}$$

\dagger see (3.49), (3.50), (3.57), (3.58), (3.55), (3.56), (3.62), (3.63), (3.69)

Figure 3.5: FBA with full scaling.

3.4 Estimation of other types of HMM's

3.4.1 Estimation of the parameters of CHMM's

The FBA still applies to the estimation problem for CHMM's [17,18,91,78]. The estimation formulas of this chapter are still valid for α , β , π_1 , and A in the case of CHMM's. Only the reestimation formulas for the output probability densities need to be revised. In the case of Gaussian mixture densities (see section 2.6) the formulas are [78]:

$$\begin{cases} \bar{\lambda}_{jm} &= \frac{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} \lambda_{jm} b_{jm}(O_t) \beta_t(j)}{\sum_{t=1}^L \alpha_t(j) \beta_t(j)} \\ \bar{\mu}_{jm} &= \frac{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} \lambda_{jm} b_{jm}(O_t) \beta_t(j) O_t}{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} \lambda_{jm} b_{jm}(O_t) \beta_t(j)} \\ \bar{C}_{jm} &= \frac{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} \lambda_{jm} b_{jm}(O_t) \beta_t(j) (O_t - \mu_{jm})(O_t - \mu_{jm})^T}{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} \lambda_{jm} b_{jm}(O_t) \beta_t(j)} \end{cases} \quad (3.87)$$

In the case of the multivariate Gaussian density the formulas simply become:

$$\begin{cases} \bar{\mu}_j &= \frac{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} b_j(O_t) \beta_t(j) O_t}{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} b_j(O_t) \beta_t(j)} \\ \bar{C}_j &= \frac{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} b_j(O_t) \beta_t(j) (O_t - \mu_j)(O_t - \mu_j)^T}{\sum_{t=2}^L \sum_{i=1}^* \alpha_{t-1}(i) a_{ij} b_j(O_t) \beta_t(j)} \end{cases}, \quad (3.88)$$

which is equivalent to:

$$\begin{cases} \bar{\mu}_j &= \frac{\sum_{t=1}^L \alpha_t(j) \beta_t(j) O_t}{\sum_{t=1}^L \alpha_t(j) \beta_t(j)} \\ \bar{C}_j &= \frac{\sum_{t=1}^L \alpha_t(j) \beta_t(j) (O_t - \mu_j)(O_t - \mu_j)^T}{\sum_{t=1}^L \alpha_t(j) \beta_t(j)} \end{cases}, \quad (3.89)$$

i.e.,

$$\begin{cases} \bar{\mu}_j &= \frac{\sum_{t=1}^L \gamma_t(j) O_t}{\sum_{t=1}^L \gamma_t(j)} \\ \bar{C}_j &= \frac{\sum_{t=1}^L \gamma_t(j) (O_t - \mu_j)(O_t - \mu_j)^T}{\sum_{t=1}^L \gamma_t(j)} \end{cases}. \quad (3.90)$$

These formulas have a similar expected-value-like (or frequency-count-like) interpretation as the one described earlier for DHMM's.

3.4.2 Estimation of the parameters of HSMM's

Expressing the likelihood function becomes somewhat cumbersome in the case of HSMM's. However it can be shown that, in the case of left-to-right models, a modified version of the forward-backward algorithm still applies to the problem of estimating $\eta = (\pi_1, A, B, D)$. The reestimation formulas are given in [88].

3.5 Evaluation of the forward-backward algorithm

3.5.1 Evaluation of the correctness of the algorithm

If the output probability matrix B is diagonal, the FBA converges in one iteration to the frequency count results of section 3.1 for matrices A and π_1 , and B remains unchanged (identity matrix). If one is only interested in the correctness of the algorithm, one can just pick a random Y sequence, run the FBA, and check that the estimated \bar{A} and $\bar{\pi}_1$ agree with the theoretical solution of formulas (3.4) and (3.6). If one is also interested in the accuracy of the estimation process, one can generate an observation sequence Y from an actual model with a diagonal B , using the Monte Carlo simulation described in the next section. An example of the first approach is shown below:

- Training sequence:

$$Y = \{1, 1, 1, 1, 1, 1, 2, 2, 2, 1, 1, 2, 2, 2, 2\}$$

- Initial estimate λ_1 :

$$A = \begin{pmatrix} 0.4 & 0.6 \\ 0.8 & 0.2 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix}$$

- Theoretical FBA solution:

$$A = \begin{pmatrix} \frac{3}{4} & \frac{1}{4} \\ \frac{1}{6} & \frac{5}{6} \end{pmatrix} = \begin{pmatrix} 0.75 & 0.25 \\ 0.167 & 0.833 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

- Practical FBA solution (after one iteration) λ_2 :

$$A = \begin{pmatrix} 0.74999 & 0.25000 \\ 0.16666 & 0.83333 \end{pmatrix} \quad B = \begin{pmatrix} 0.999 & 0.000 \\ 0.000 & 1.000 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.9999 \\ 0.0000 \end{pmatrix}$$

To further check the correctness of the estimation (for B in particular), one can use the theoretical FBA results of section 3.2.3.2 for a very simple model, and compare them with the practical results of the FBA. Using the same notation as in section 3.2.3.2 we define: $F^{(\ell)} = (f_{ij}(\ell))$, $H^{(\ell)} = (h_{ik}(\ell))$, and $P^{(\ell)} = P(X^{(\ell)}, Y/\lambda)$. This notation is used in the following example:

- Training sequence: $Y = \{1, 2, 1\}$

- Initial estimate λ_1 :

$$A = \begin{pmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{pmatrix} \quad B = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix}$$

- Frequency counts:

$$\ell = 1 \quad X^{(1)} = \{1, 1, 1\} \quad P^{(1)} = 0.0376 \quad F^{(1)} = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \quad H^{(1)} = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\ell = 2 \quad X^{(2)} = \{1, 1, 2\} \quad P^{(2)} = 0.0013 \quad F^{(2)} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \quad H^{(2)} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\ell = 3 \quad X^{(3)} = \{1, 2, 1\} \quad P^{(3)} = 0.0071 \quad F^{(3)} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad H^{(3)} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\ell = 4 \quad X^{(4)} = \{2, 1, 1\} \quad P^{(4)} = 0.0018 \quad F^{(4)} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad H^{(4)} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\ell = 5 \quad X^{(5)} = \{1, 2, 2\} \quad P^{(5)} = 0.0015 \quad F^{(5)} = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \quad H^{(5)} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

$$\ell = 6 \quad X^{(6)} = \{2, 2, 1\} \quad P^{(6)} = 0.0020 \quad F^{(6)} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \quad H^{(6)} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

$$\ell = 7 \quad X^{(7)} = \{2, 1, 2\} \quad P^{(7)} = 0.0001 \quad F^{(7)} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad H^{(7)} = \begin{pmatrix} 0 & 1 & 0 \\ 2 & 0 & 0 \end{pmatrix}$$

$$\ell = 8 \quad X^{(8)} = \{2, 2, 2\} \quad P^{(8)} = 0.0004 \quad F^{(8)} = \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} \quad H^{(8)} = \begin{pmatrix} 0 & 0 & 0 \\ 2 & 1 & 0 \end{pmatrix}$$

- This leads to the following matrices:

$$(n_{ij}) = \begin{pmatrix} 0.0783 & 0.0100 \\ 0.0110 & 0.0043 \end{pmatrix} \quad (m_{ik}) = \begin{pmatrix} 0.0960 & 0.0408 & 0 \\ 0.0076 & 0.0110 & 0 \end{pmatrix} \quad (q_i) = \begin{pmatrix} 0.0475 \\ 0.0043 \end{pmatrix}$$

- Theoretical FBA reestimates from section 3.2.3.2:

$$A = \begin{pmatrix} 0.89 & 0.11 \\ 0.72 & 0.28 \end{pmatrix} \quad B = \begin{pmatrix} 0.70 & 0.30 & 0 \\ 0.41 & 0.59 & 0 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.92 \\ 0.08 \end{pmatrix}$$

- Practical FBA results (after one iteration) λ_2 :

$$A = \begin{pmatrix} 0.887118 & 0.112881 \\ 0.713315 & 0.286684 \end{pmatrix} \quad B = \begin{pmatrix} 0.701677 & 0.298322 & 0 \\ 0.410025 & 0.589974 & 0 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.916988 \\ 0.083012 \end{pmatrix}$$

3.5.2 Evaluation of the modelling capabilities of the algorithm

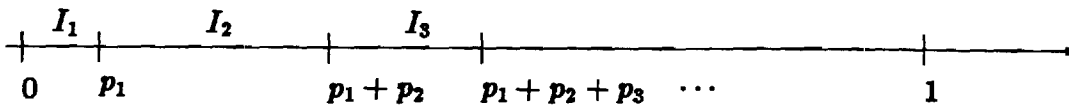
Given an HMM $\lambda = (\pi_1, \mathcal{A}, B)$ one can generate a sequence of states $\{X_t\}$, and the corresponding sequence of observations $\{Y_t\}$ with a Monte Carlo simulation (see below). The sequence $\{Y_t\}$ can then be used as a training sequence on which the FBA is applied. It is therefore possible to determine how good the FBA is at recovering the original model λ (influences of the initial estimate, training sequence length ... can be studied). The Monte Carlo simulation can be simply described: let S be a source which can produce letters from the alphabet $\mathcal{A} = \{1, 2, \dots, N\}$ with probabilities $\{p_1, p_2, \dots, p_N\}$ such that $\sum_{i=1}^N p_i = 1$. Our intention is to generate a sequence $S = \{s_1, s_2, \dots, s_L\}$ of letters from \mathcal{A} that will simulate the behavior of the source S — in other words, the sequence S will look as if it had actually been generated by the source S . For this purpose, let Z be a random variable uniformly distributed in $[0, 1]$ (and simulated in our experiment by a random number generator). A sequence $\{z_1, z_2, \dots, z_L\}$ of outcomes of Z can simply be mapped into a sequence $\{s_1, s_2, \dots, s_L\}$ because of the following properties:

$$\forall 0 \leq a \leq 1 \quad \forall 0 \leq b \leq 1 \quad a < b \quad a + b \leq 1 \quad \Pr(a < Z \leq a + b) = b, \quad (3.91)$$

and if we recursively generate the following sequence of intervals I_n :

$$\begin{aligned} I_1 &= [0, p_1] \\ I_n &=]\ell_n, r_n] \quad \text{with} \quad \ell_n = r_{n-1}, \quad r_n = \ell_n + p_n, \end{aligned}$$

i.e., a sequence of intervals of the form:



It follows from (3.91) that

$$\Pr(Z \in I_n) = p_n, \quad (3.92)$$

i.e., the event “Z falls in the interval I_n ” occurs with the same probability as the event “the letter n was drawn.” Therefore the Monte Carlo simulation generates a random number Z , and the letter n , when Z is in I_n . This procedure can be applied to generate the first state i_1 , based on the probabilities π_1 , then the first observation k_1 , based on the probabilities $b_{i_1 k}$, then the next state i_2 , based on the probabilities $a_{i_1 i}$, then the observation k_2 , based on the probabilities $b_{i_2 k}$, etc. An example of a Monte Carlo simulation is given below:

- Original model λ :

$$A = \begin{pmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{pmatrix} \quad B = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix}$$

From this model, a state sequence (starting in state 1), and an observation sequence, both of length 500, were generated. Then the FBA was run for 99 iterations with the following random initial estimates:

- Initial estimates λ_1 :

$$A = \begin{pmatrix} 0.255 & 0.745 \\ 0.092 & 0.908 \end{pmatrix} \quad B = \begin{pmatrix} 0.177 & 0.382 & 0.441 \\ 0.204 & 0.442 & 0.354 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 0.396 \\ 0.604 \end{pmatrix}$$

- The estimated model λ_{100} was the following⁴:

$$A = \begin{pmatrix} 0.78 & 0.22 \\ 0.42 & 0.58 \end{pmatrix} \quad B = \begin{pmatrix} 0.70 & 0.14 & 0.16 \\ 0.11 & 0.39 & 0.50 \end{pmatrix} \quad \pi_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Even though the initial estimate is significantly different from the actual model, the estimated model is a good approximation of the actual model. The quality of the estimation is a function of not only the FBA, but also the value of the initial

⁴Actually the FBA performed a state permutation: what the original model called state 1, the FBA called it state 2. For the results shown here, the states have been permuted back to their original index name.

estimate, the training sequence length, and the quality of the random number generator.

CHAPTER 4

Maximum likelihood trellis decoding

4.1 Presentation of the problem

Being given an HMM $\lambda = (\pi_1, A, B)$ (estimated with the procedure described in chapter 3) one might want to perform two inverse operations on specific data. Most of the time these operations will be called *coding* and *decoding* (referring to the emitter/receiver communication model of chapter 1):

- given an observation sequence $Y(1:L)$ we would like to retrieve the underlying, but hidden, structure of Y by recovering the corresponding state sequence $X(1:L)$. This process which might occur in a speech coder at the emitter is called the (state) coding. The terms (state) encoding and (state) decoding or emitter decoding might also be used since the goal is to decode the states from the fuzzy observed Y sequence.
- inversely, given a state sequence $X(1:L)$ we would like to regenerate the corresponding observation sequence $Y(1:L)$ with minimum distortion. From the communication model point of view this will be called the (observation) decoding process. Another term is receiver decoding.

Let us consider the state-decoding problem. Finding X_n , the state of the system at time n , is not straightforward because speech goes through a double chaining process: one for the states, and one for the observations. The state X_n

depends on the past states $X(1:n-1)$ and the future states $X(n+1:L)$. The same is true for the observations. These dependencies have been modelled by 1st order Markov chains and are summarized in the model λ . An optimum selection of X_n cannot be made at time n only; a global optimum selection has to be made based on the sequences X and Y over a period of time. Consistent with the estimation method of chapter 3, a maximum likelihood decoding will be used. The optimum state sequence X is the one which maximizes the joint probability of X and Y . As seen in chapter 3 the joint probability of $X^{(\ell)}$ and Y is:

$$P(X^{(\ell)}, Y) = P(X^{(\ell)})P(Y/X^{(\ell)}),$$

where $P(X^{(\ell)})$ and $P(Y/X^{(\ell)})$ are respectively given in formulas (3.9) and (3.15). From now on the superscript (ℓ) identifying a specific state sequence will be dropped. The joint probability takes the form:

$$P(X, Y) = \prod_{n=1}^L a_{x_{n-1}x_n} b_{x_n}(Y_n). \quad (4.1)$$

Maximizing the probability $P(X, Y)$ is equivalent to maximizing the a posteriori probability $P(X/Y)$ since $P(X, Y) = P(X/Y)P(Y)$ and Y is given. Therefore the ML decoding is a Maximum a Posteriori (MAP) method, which is also known to minimize the probability of error.

The ML decoding problem is therefore to maximize a function \mathcal{F} over the discrete set $\{x_i\}$ where \mathcal{F} has the following form:

$$\mathcal{F}(X, Y) = \prod_{n=1}^L G(x_{n-1}, x_n) F(x_n, Y_n), \quad (4.2)$$

and where the functions F and G are in $[0,1]$ and Y is given. Even though \mathcal{F} is a finite product of positive terms, the maximization of \mathcal{F} is not straightforward because the term at time n depends on the term at time $n-1$ and therefore all the terms in the product are interdependent. The expression for \mathcal{F} is quite general, and will apply to many types of decoding algorithms. In particular: the state

and the observation decoding problems, and the decoding for hidden semi-Markov models.

Two interesting interpretations of the function \mathcal{F} should be mentioned. They are better dealt with in their logarithmic form. Using the same convention as in chapter 3 ($\hat{a} = \log_{10} a$), we have:

$$\hat{\mathcal{F}}(X, Y) = \sum_{n=1}^L [\hat{G}(x_{n-1}, x_n) + \hat{F}(x_n, Y_n)]. \quad (4.3)$$

We define:

$$\mathbf{L}_L(X, Y) = -\hat{\mathcal{F}}(X, Y) \quad (4.4)$$

$$\ell_n = -[\hat{G}(x_{n-1}, x_n) + \hat{F}(x_n, Y_n)] \quad (4.5)$$

$$\mathbf{L}_L(X, Y) = \sum_{n=1}^L \ell_n. \quad (4.6)$$

\mathbf{L} can be computed recursively:

$$\mathbf{L}_n(X, Y) = \mathbf{L}_{n-1}(X, Y) + \ell_n. \quad (4.7)$$

$\mathbf{L}(X, Y)$ is called the log-likelihood function and takes values in $[0, +\infty[$. The log-likelihood over a single state sequence, \mathbf{L} , is not to be confused with the log-likelihood over all possible state sequences, \mathcal{L} , encountered in chapter 3. As a first interpretation, $\mathbf{L}(X, Y)$ is seen as a measure of the “distance” between the two sequences X and Y . The better the XY match, the shorter the distance $\mathbf{L}(X, Y)$. As a second interpretation, $\mathbf{L}(X, Y)$ represents the length (or weight) of a state-node path through the trellis of the HMM. The better the X sequence, the shorter (resp. the lighter) the path length (resp. the weight) $\mathbf{L}(X, Y)$. As a relative measure of the closeness of the two sequences X and Y , the log-likelihood \mathbf{L} enables us to compare how two different state sequences $X^{(1)}$ and $X^{(2)}$ relate to a given observation sequence Y . The ML decoding problem is equivalent to finding the most likely sequence X given Y (maximize the likelihood \mathcal{F}), or finding the X which minimizes the distance $\mathbf{L}(X, Y)$ (minimize the log-likelihood \mathbf{L}), or finding

the shortest path $X(1:L)$ (minimize the length $L_L(X, Y)$).

This interpretation leads us to the concept of a search through the HMM trellis to find the optimum (shortest) path X given Y . This formulation allows a recursive optimization using a dynamic programming technique [19,67]. Such an algorithm, known as the *Viterbi algorithm*, was initially introduced in the field of digital communications for the decoding of convolutional codes [20,56,102,145,146]. The X sequence was a sequence of binary digits transmitted from the emitter over a communication channel. A noisy observation of X , the sequence Y , was detected at the receiver. The sequences X and Y were of the same nature. The Hamming distance (mainly a count of the number of times the two sequences differed over a given finite period of time) was used as a measure of closeness. Recovering the X 's from the noisy Y 's was achieved by minimizing the Hamming distance in the network of all possible state paths. In the case of the speech waveform the state and observation sequences are not similar in nature and the measure of closeness is a probabilistic distance. However the same concept of search through a graph (or network) is applicable.

4.2 The trellis decoding algorithm

4.2.1 The algorithm

The trellis decoding algorithm (TDA), or Viterbi algorithm, finds the optimum sequence $X(1:L)$ as the shortest path through the trellis of the HMM. A recursive search through the trellis, based on a dynamic programming technique, finds the best path without performing an exhaustive search of all the possible paths. First of all, before describing the algorithm, let us introduce a small useful vocabulary applying to the concept of trellis search:

- **node:** the trellis is made of successive layers of N nodes; N is the node alphabet size; there is one layer for each time index n ; the nodes might represent states (coding process) or observations (decoding process). See node $\{a\}$ in figure 4.1.
- **branch:** a line connecting 2 nodes in 2 consecutive layers; the length of the branch $\{x_{n-1}, x_n\}$ is ℓ_n . See branch $\{ab\}$ in figure 4.1.
- **path:** a concatenation of consecutive branches; the length $L_n(X, Y)$ of a path is the sum of the lengths of its branches. See path $\{abc'd'e\}$ in figure 4.1.
- **depth:** the number of layers in the path $X(1:L)$, i.e., the sequence length L — not to be confused with the corresponding path length $L_L(X, Y)$. The depth of the path $\{abc'd'e\}$ is 5.
- **extension:** a branch connected to the last node of a path to increase its depth by 1. See extension $\{ef\}$ in figure 4.1.
- **parent node:** the node on the previous layer $n - 1$ to which a node in the present layer n is connected (also called a predecessor). A node X_n can have only one parent $\mathcal{P}(X_n)$. See $\{a\}$ as the parent of $\{b\}$ in figure 4.1.
- **child node:** a node which has another node for parent (also called a successor). A node can have several children. See nodes $\{b, b', b'', D\}$ as the children of $\{a\}$ in figure 4.1.
- **dead node:** a node without any children. See node $\{D\}$ in figure 4.1.
- **alive node:** a node which is not dead.
- **survivors:** the survivors of depth n are the N shortest paths of depth n , each one of them ending in a different node in layer n . If there are several paths of exactly the same length L_n ending in the same node in layer n , one path is selected at random. Each survivor will usually be identified, at a given

time, by an index which is also the node in which the survivor ends (survivor i ends in node i).

- backtracking: starting from a node X_n in layer n and finding the sequence $\{P(X_n), P(P(X_n)), \dots, P \dots P(P(X_n)) \dots\} = P(X_2)\}$.
- $G(x_1) = G(x_0, x_1)$ is the initial distribution of the nodes (same convention as in chapter 3 for the initial distribution of the states).
- $G(x_{n-1}, x_n)$ is the probability of transition between the nodes x_{n-1} and x_n in the trellis.
- $F(x_n, Y_n)$ is the probability of observing the quantity Y_n while in node x_n .

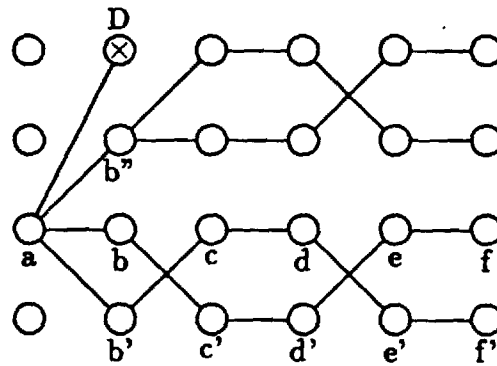


Figure 4.1: Example of a trellis.

Let us now justify how the trellis decoding algorithm operates:

- **goal:** find the shortest path of depth L , $X(1:L)$.
 - such a path exists since there is a finite denumerable number (N^L) of possible paths; one is the shortest; in case of ties one path is selected at random.
 - this path ends in the node X_L (see figure 4.2). It is, by definition, the shortest of the N survivors of depth L (see definition of survivors above).
- **sub-goal 1:** find the N survivors of depth L .
 - the process is identical for each of the N survivors. Let us find the survivor of depth L ending in node $X_L = 1$ (see figure 4.3).
 - this survivor of depth L had to be extended from a path of depth $L - 1$ ending in node x_{L-1} . This path of depth $L - 1$ ending in x_{L-1} had to be the shortest of such paths (i.e. had to be the survivor of depth $L - 1$ ending in x_{L-1}), otherwise a shorter path of depth $L - 1$ ending in x_{L-1} , when extended, would have led to a "shorter survivor" of depth L ending in x_L (which is not possible by definition of a survivor), since these two paths of depth $L - 1$ have the same length extension. Therefore one must find the survivor of depth $L - 1$ ending in x_{L-1} .
 - however x_{L-1} is not actually known, therefore one has to find all the N survivors of depth $L - 1$ and compute the N extensions of these N survivors to the node 1 (see figure 4.4). The shortest extended path is selected: it is the desired survivor of depth L ending in node 1 (and x_{L-1} is now known).
- **sub-goal 2:** find the N survivors of depth $L - 1$, etc,
 - i.e., iterate sub-goal 1 from $n = L$ to $n = 2$.

Therefore the trellis decoding algorithm can be summarized as follows (see also figure 4.5):

- find the N survivors for every depth from $n = 2$ to $n = L$ recursively (forward move).
- find the best survivor of depth L by backtracking from the last node (backward move).

The corresponding algorithm in pseudo-code is shown in figure 4.6. The following notations have been used:

- t : time index of a layer
 j : a node in the current layer at time t
 i : a node in the previous layer at time $t - 1$
 $P_t(j)$: the parent node, in layer $t - 1$, of the node j , in layer t
 $Cur_L(j)$: length of the survivor of depth t ending in node j
 $Pre_L(i)$: length of the survivor of depth $t - 1$ ending in node i
 $L(i)$: pseudo-length of the path of depth t , extended, by the branch $i \rightarrow j$ (j given), from the survivor of depth $t - 1$ ending in node i .
 $Imin$: a function that returns the index where the minimum was reached:
 $i_0 = Imin_i[f(i)] \iff \forall i \quad f(i_0) \leq f(i)$
 "I" is the index operator acting on the function min.

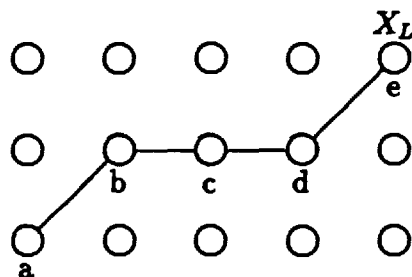


Figure 4.2: The shortest path of depth 5.

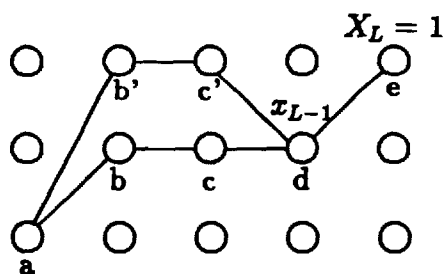


Figure 4.3: The shortest path is the extension of a survivor of depth 4.

$$L\{abcde\} < L\{ab'c'de\} \Leftrightarrow L\{abcd\} < L\{ab'c'd\}$$

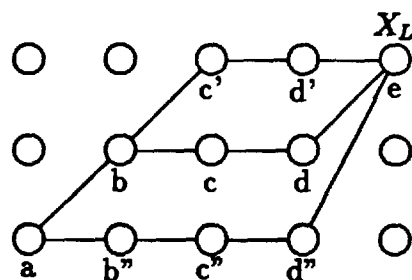


Figure 4.4: Selecting the best path extending to X_L .

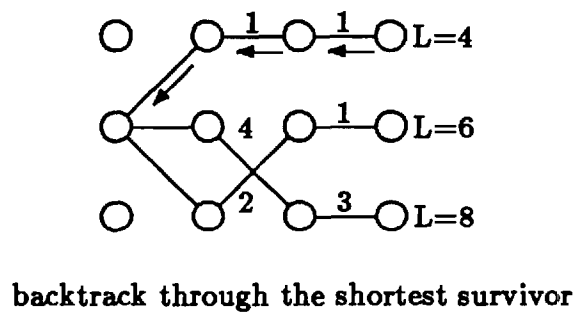
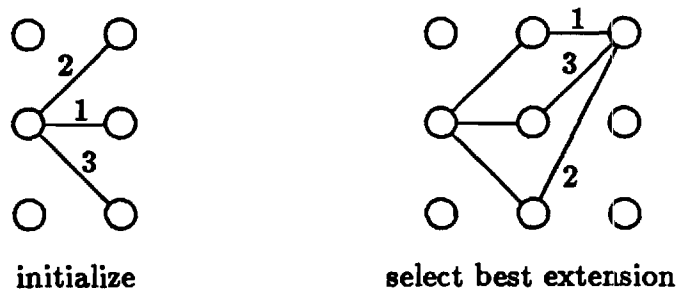


Figure 4.5: Summary of the TDA procedure (numbers are branch lengths).

4.2.2 The issue of the initial node

In the trellis algorithm of figure 4.6 we distinguished between the initial node being known and unknown. Such a distinction can be avoided in the algorithm. If the initial node is known, the N survivors of depth 2 and their lengths are known too. Then the algorithm starts solving for the survivors of depth 3. In this case, the term $-\hat{G}(X_1) - \hat{F}(X_1, Y_1)$, appearing in the algorithm of figure 4.6, is constant for the N survivors and does not affect the minimization of the overall lengths. It could be dropped, but is usually kept for consistency with the case of the unknown initial node. If the initial node is unknown the algorithm starts solving for the survivors of depth 2 by considering the N possible starting nodes, computes the

```

If  $X_1$  known then                                ;initialization
    For  $j = 1, N$ 
         $\mathcal{P}_2(j) = X_1$ 
         $Pre\_L(j) = -\hat{G}(X_1) - \hat{F}(X_1, Y_1) - \hat{G}(X_1, j) - \hat{F}(j, Y_2)$ 
    End For  $j$ 
     $t_0 = 3$ 
Else
    For  $j = 1, N$ 
         $Pre\_L(j) = -\hat{G}(j) - \hat{F}(j, Y_1)$ 
    End For  $j$ 
     $t_0 = 2$ 
End if
For  $t = t_0, L$                                     ;trellis depth loop
    For  $j = 1, N$ 
        For  $i = 1, N$ 
             $L(i) = Pre\_L(i) + \hat{G}(i, j)$                 ;extend lengths
        End For  $i$ 
         $\mathcal{P}_i(j) = \text{Imin}_i[L(i)]$                         ;find parent node
         $Cur\_L(j) = L[\mathcal{P}_i(j)] + \hat{F}(j, Y_i)$             ;update lengths
    End For  $j$ 
    For  $j = 1, N$ 
         $Pre\_L(j) = Cur\_L(j)$ 
    End For  $j$ 
End For  $t$ 
 $X_L = \text{Imin}_i[Cur\_L(i)]$ 
 $X = X_L$ 
For  $t = L, 2$                                         ;backtracking
     $X_{t-1} = \mathcal{P}_t(X)$ 
     $X = X_{t-1}$ 
End For  $t$ 

```

Figure 4.6: The basic ML trellis decoding algorithm.

lengths of the corresponding branches and selects the shortest branches.

These two cases (initial node known and unknown) can be condensed into a single case controlled and parameterized by the initial distribution of the nodes $G(X_1)$. The fact that the initial node (say i_0) is known is equivalent to having:

$$\begin{cases} \hat{G}(i) = 0 & \forall i \neq i_0 \\ \hat{G}(i_0) = 1. \end{cases} \quad (4.8)$$

Running the algorithm as if the initial node were unknown, with the above initial distribution of the nodes, is equivalent to running the algorithm with the known starting node i_0 . The branches not starting with i_0 will have an infinite length; minimizing the lengths will be equivalent to choosing i_0 as the starting node. In practice zero entries in F and G would lead to infinite values for \hat{F} and \hat{G} , therefore the entries in $F(i, k)$ and $G(i, j)$ are constrained to be greater than a strictly positive lower bound 10^{-R} (this automatically happens when F and G are the results of the training process described in chapter 3). For the initial distribution of the nodes $G(i)$, zero entries or entries lower than a given lower bound (meaning they have to be considered 0) are allowed, to enable the selection of the initial node(s) through $G(i)$. But then the corresponding logarithm $-\hat{G}(i)$ has to be set to 10^{R-T} (the equivalent of $+\infty$). T is a security margin such that, when several of these "infinite" lengths are added together, they do not exceed 10^R (the computer range; see chapter 3). A path with an "infinite" length will actually never be selected (there will always be a path with a shorter length) therefore we do not need to worry about having to add up such large lengths, and T can be (almost) 0¹. When it comes to adding up finite lengths it is very improbable that the total length will exceed the computer range (unless the computer range is very small, or the sequence length L is very large). This is because:

¹Only one extension to an "infinite" length will be computed. We only need to ensure: $10^{R-T} + R \leq 10^R$, i.e., $T = R - \log_{10}[10^R - R] \approx 0$.

$$\begin{aligned}\sum_{n=1}^L [-\hat{G}_n - \hat{F}_n] &\leq \sum_{n=1}^L [\max(-\hat{G}_n) + \max(-\hat{F}_n)] \\ &\leq \sum_{n=1}^L 2R\end{aligned}$$

$$L_L(X, Y) \leq 2RL.$$

This is no problem as long as $L \leq \frac{10^R}{2R}$ (this condition will be met in practice since $R = 74$ and $L = 60,000$). In the rare case where this condition is not met, the solution is to subtract a constant from all the lengths at a proper time n , to bring down the lengths to manageable size (this would not of course influence the relative comparison of lengths). The condensed algorithm driven by the initial distribution of the nodes for the choice of the starting node(s) is shown in figure 4.7.

For speech processing applications is the initial node known? When concerned with the state-decoding problem, the initial node is usually unknown. In that case, the algorithm should be started either by picking a starting state at random², or by initializing the initial distribution of the states with the steady-state distribution of the states. However, there is one situation when the starting state is exactly known: this is when the state decoding has to be performed on the observation sequence with which the HMM was trained. The initial node X_1 is the only node x_1 for which the initial distribution of the states is not zero $G(X_1) \neq 0$. There is another situation when the initial state is fairly well known: this is when the sequence to decode starts with silence/noise, not speech. As will be seen in chapter 6, the initial state can then be selected from a small number of possible silence/noise states. Moreover, the selection of the correct initial node is not critical for silence.

When dealing with the observation-decoding process, the initial observation is always known (it is the first LPC vector of the speech waveform), but might not be available at the receiver (if it was not transmitted). Usually one would transmit

²One might also choose the state X_1 which maximizes $b_x(Y_1)$ over x .

```

D = 10;  R = 74;  T = 1
For  $j = 1, N$                                      ;initialize
    If  $G(j) < 10^{-D}$  then
         $\hat{G}(j) = -10^{R-T}$ 
    Else
         $\hat{G}(j) = \log_{10}[G(j)]$ 
    End If
     $L(j) = -\hat{G}(j) - \hat{F}(j, Y_1)$ 
End For  $j$ 
For  $t = 2, L$                                      ;trellis depth loop
    For  $j = 1, N$ 
         $E_m = 10^R$ 
        For  $i = 1, N$ 
             $E(i) = L(i) - \hat{G}(i, j)$                  ;extend lengths
            If  $\min(E(i), E_m) = E(i)$  then
                 $p_i(j) = i$                          ;find parent node
                 $E_m = E(i)$ 
            End If
        End For  $i$ 
         $\tilde{L}(j) = E_m - \hat{F}(j, Y_t)$                  ;update lengths
    End For  $j$ 
    For  $j = 1, N$ 
         $L(j) = \tilde{L}(j)$ 
    End For  $j$ 
End For  $t$ 
 $X_L = \text{Imin}_i[L(i)]$ 
 $X = X_L$ 
For  $t = L, 2$                                      ;backtracking
     $X_{t-1} = p_t(X)$ 
     $X = X_{t-1}$ 
End For  $t$ 

```

Figure 4.7: Optimum, unconstrained ML trellis decoding algorithm.

the first observation, otherwise a random selection would have to be made³.

In any case the possibility of the indetermination of the initial node is easily overcome because the trellis algorithm behaves smoothly when it searches for a global optimum path. A wrong initial node will only possibly lead to short and transient wrong decisions over a few nodes at the beginning of the path, which will be rapidly smoothed out by the influence of the global decoding over the upcoming layers. In other words the trellis decoding algorithm resynchronizes itself quickly: it may start with a few unreliable estimates of the first nodes (generally silence and noise, not speech), but normal decoding will prevail thereafter.

4.2.3 Algorithm complexity

A direct minimization of the overall log-likelihood by examining all the possible paths is not feasible because there are N^L such paths. This number is exponentially growing with the observed sequence length L . The base of the exponential is the number of nodes N (whether 64 or 1024). The trellis decoding algorithm, even though it considers only a small subset of all the possible paths, is optimum in the sense that it finds the exact solution to the maximum likelihood decoding problem.

A tremendous economy is achieved because the algorithm keeps only a constant number of paths alive over time (the N survivors). The number of survivors is linear with the number of nodes N , and is independent of the trellis depth L . For each layer, and for each node in this layer, only N additions have to be performed to compute the lengths of the extending branches. An extra addition per node and per layer has to be carried out to update the N survivor lengths⁴. Globally the number of additions required is $N(N+1)L - N^2$. The computational

³One might also select the observation Y_1 which maximises $b_{X_1}(y)$ over y .

⁴The matrices F and G are stored in logarithmic form, no extra log-computations are required.

complexity is linear in L (no longer exponential in L), of the order N^2L . Moreover the algorithm is very well suited for multi-processor parallel implementations. A single simple processor can be assigned to each one of the N nodes. Each processor only needs to compute N branch extension lengths (N additions), compare the extension lengths two by two, and generate the new survivor length (1 addition). However, the great computational efficiency of the trellis decoding algorithm is partly gained at the price of a large memory requirement. In every layer the parent of each node needs to be stored, which requires $N(L - 1)$ storage locations. About 16 megabytes for $N = 64$, $L = 60000$ on the basis of 4 bytes per entry. Two additional N dimensional arrays of storage are needed for the survivor lengths for a total memory storage of $BN(L + 1)$ (B is the number of bytes per entry).

Hopefully keeping track of the optimum path over 60000 layers is not what one really wants. Optimizing the operation of the decoder over a few milli-seconds or even a few seconds, instead of a few minutes, is more likely to be what we are looking for. Therefore the actual L will be much smaller and the required memory much more manageable. Tables 4.1 and 4.2 show the computation and memory costs of the state-decoding trellis algorithm ($N = 64$) as a function of the duration of the speech (on a basis of 15ms speech frames). The issue of the memory cost will be further examined in the next section about convergence nodes.

Table 4.1: Computational cost of the state-decoding trellis algorithm.

Number of frames	Speech duration	Number of additions	CPU time		
			IBM-PC	MV/10000	Cray 1
10	150ms	37,504	0.38s	12.5ms	0.23ms
15	225ms	58,304	0.58s	19.4ms	0.36ms
20	300ms	79,104	0.79s	26.3ms	0.49ms
30	450ms	120,704	1.21s	40.2ms	0.75ms
60	900ms	245,504	2.42s	81.4ms	1.50ms
80	1.2s	328,704	3.29s	0.11s	2.05ms
100	1.5s	411,904	4.12s	0.14s	2.57ms
130	1.95s	536,704	5.37s	0.18s	3.35ms
200	3.0s	827,904	8.24s	0.28s	5.14ms
330	4.95s	1,368,704	13.69s	0.46s	8.55ms
4,000	1mn	16,635,904	2m46s	5.55s	0.11s
60,000	15mn	249,595,904	41m35s	1m23s	1.56s

Table 4.2: Memory cost of the state-decoding trellis algorithm.

Number of frames	Speech duration	memory (B=1)	memory (B=4)†
10	150ms	0.7kb	2.8kb
15	225ms	1.0kb	4.1kb
20	300ms	1.3kb	5.4kb
30	450ms	2.0kb	7.9kb
60	900ms	3.9kb	15.6kb
80	1.2s	5.2kb	20.7kb
100	1.5s	6.5kb	25.8kb
130	1.95s	8.4kb	33.5kb
200	3.0s	12.9kb	51.5kb
330	4.95s	21.2kb	84.7kb
4,000	1mn	256.1kb	1.1Mb
60,000	15mn	3.8Mb	16Mb

†B is the number of bytes. Actually $B = 1$ (8 bits) can represent at most 256 nodes; in practice no more than 10 bits (1024 nodes) will be needed.

4.3 The concept of convergence nodes

Still it appears from the preceding section that the memory needed to decode only a few seconds of speech is quite significant. Would it be possible to achieve an optimum, or near-optimum, decoding of shorter segments of speech? Would it be possible to divide and group long speech sequences into blocks of K trellis layers (or frames)? Could we then perform the optimum decoding on the K layers and then concatenate the corresponding survivors of depth K to build the overall "optimum" solution? As will be seen later this is somewhat loosely formulated. Is there a value of K , much less than L , that would allow optimum decoding? Could one keep K below, say 10, to achieve optimum decoding? This factor K will be given different names: the convergence length, the pegging period, or the decoder delay (in the sense that the decoder has to wait for K frames of speech before being able to generate the optimum decoded sequence). All these denominations are more or less equivalent. The convergence length is an important factor that will reflect the degree of predictability of the speech, as well as the long term inter-frame dependencies of the speech. The value(s) of K will determine the block length of speech segments which are totally independent of past and future segments. This concept will become clearer as we move on. One of the goals of the HMM is to capture the long term (several frames) predictability of speech, as opposed to the short term (several samples, one frame) and the very long term (several tens of frames) predictability of speech. The convergence length is therefore of theoretical and practical importance in the following three areas:

- memory requirement for the trellis decoding algorithm,
- speech predictability over time,
- decoder delay.

But how does the convergence length relate to the actual inner-workings of the trellis decoding algorithm (outside from a desirable feature over-imposed on

the algorithm itself)? The next paragraphs will provide the beginning of an answer and make clear why the names “convergence length” and “pegging period” have been chosen.

As a matter of fact, the idea of grouping trellis layers into independent blocks is directly related to an interesting, usually spontaneous, feature of the trellis decoding algorithm. This feature can very often be observed when following the behavior of the algorithm over time on actual data. The N survivors are recursively generated during a forward move. Their depth increases by 1 every time a new layer is reached. Their lengths steadily increase. An interesting image is to look at the survivors as moving forward, pointing their “heads” in the direction of increasing time indices. Some of the survivor lengths will increase much faster than others, and the discrepancy between some of the survivors will be quite significant. As time increases some of the survivor lengths will be so much larger than other survivor lengths, that the former survivors will have no hope of catching up and being selected as the shortest survivor over the upcoming layers, and finally they will die (end up in a dead node). It turns out that it usually happens that all the survivors, except one, die at a given time n . Or more precisely said, all the nodes in layer n die, except one — the convergence node. Node $\{e\}$ in figure 4.8 is a convergence node. Two of the paths which were survivors in layer 5 are no longer survivors in layer 6: two survivors died in layer 5 and were replaced by two new survivors in layer 6 (see figure 4.8). The existence of the convergence node $\{e\}$ at time n can only be detected at time $n + 1$, when all the nodes in layer $n + 1$ are found to have the same unique parent. More precisely, a convergence node at time n can only be detected at time $n + w$ ($w \geq 1$). The extra-delay w will be called the *weight* of the convergence node⁵. Considering the heads of the survivors at time $n + w$ and looking w layers backwards, it appears that the N survivors emerge from

⁵We might want to distinguish two basic types of convergence nodes: the convergence nodes of weight 1 (the simplest type), and the convergence nodes of weight greater than 1.

a single node, the convergence node. Said differently, the N survivors of depth $n + w$ have a *common tail*, the optimum path over the time range $[1, n]$ (see path $\{abcde\}$ in figure 4.8). This phenomenon of the convergence node usually appears quasi-periodically. The quasi-period of convergence is K , the convergence length. It might be a function of time, K_n . In other words the *convergence length* is the number of layers in between two convergence nodes (the two boundary layers being included). In the example of figure 4.8, $K = 5$. The term “convergence length” is used when the phenomenon happens naturally during a free run of the algorithm. It might happen that natural convergence did not occur (or did not occur before a preset maximum allowable convergence length K_m). Then a non-spontaneous convergence can be forced on the algorithm. The N survivors are forced to go through a given node at a given time, and the decoded block is transmitted by the decoder. Forcing a (presumably known) node on the survivors is called *pegging* and K is then also referred to as the *pegging period*.

We just described the concept of convergence node. The convergence was total, in the sense that all the survivors emerged from a single node. However, partial convergence might also occur: all the survivors might emerge from N_A (alive) nodes ($N_A \leq N$). N_A is the *order of the convergence*. A (total) convergence is a convergence of order 1; a partial convergence is a convergence of order greater than 1. The order of convergence is a function of time, for a given layer: it depends on the layer from which the convergence is looked at. If, seen from layer t , the order of convergence is N_A , it might appear as $N'_A \leq N_A$, from layer $t + 1$. That is, the order of convergence is a decreasing function of time, for a given layer. In figure 4.9, the order of convergence of layer 3 is 2, seen from layer 4, and 1 seen from layer 7. Now, let us describe the algorithm that detects convergence nodes. It is done through iterative partial backtrackings in the trellis. The creation of a convergence node of weight 1, automatically changes all the nodes between this convergence node and the previous convergence node, into convergence nodes of

weight greater than 1. In fact, any node on the path between two convergence nodes (referred as the initial and final convergence nodes), becomes a convergence node itself, of weight greater than the weight of the final convergence node. This is depicted in figure 4.9: $\{B\}$ is detected as a convergence node of weight 1, then nodes $\{a\}$, $\{b\}$, and $\{c\}$ become convergence nodes of respective weights 4, 3, and 2. The solution path over this period of time is now known: it is necessarily $\{A, a, b, c, B\}$. Once $\{B\}$ has been detected, the algorithm does not need to look for convergence nodes of weight greater than 1, prior to $\{B\}$. Now, will convergence nodes of weight greater than 1, ever be detected before convergence nodes of weight 1? The answer is yes. An example is shown in figure 4.10: node $\{C\}$ is detected as a convergence node of weight 2, and no convergence node of weight 1 was detected in the layer $t = 4$. Therefore the TDA should detect convergence nodes by iteratively backtracking, from the present time, down to the time layer where a new convergence node is found, or down to the layer at a time distance w from the layer of the previous convergence node (of weight w). In each layer, the algorithm looks for a convergence node of increasing weight, starting at $w = 1$. As a summary, the detection works like this: let us suppose that a convergence node of weight w was detected in layer t ($t = 0$ for convenience). The TDA reached layer w (from where the convergence node was detected by backtracking). The TDA keeps moving forward, one layer at a time, and looks for new convergence nodes. At layer $t = w + 1$ it looks for a convergence node of weight 1. If there is no such node, the TDA proceeds in layer $t = w + 2$, looks first for a convergence node of weight 1, then if unsuccessful, of weight 2. Then the TDA moves forward to layer $t = w + 3$, etc ... The actual decoding delay D is the sum of the convergence length K and the weight w : $D = K + w$. After a given convergence node, D layers have to be processed to detect the new convergence node. Then, the solution path over the K layers can be "output" by the algorithm.

The expression "optimum path" was used in the preceding paragraph to describe the common tail. Is it appropriate? Well, yes and no. It is appropriate

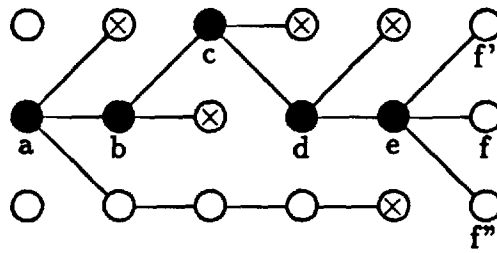


Figure 4.8: Emergence of the convergence node $\{e\}$.

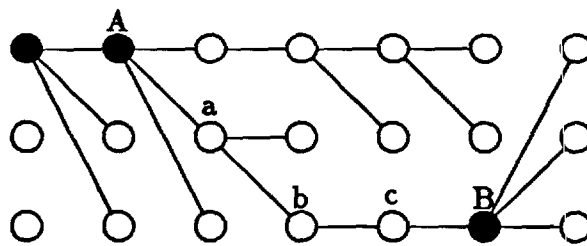


Figure 4.9: The solution path $\{A, a, b, c, B\}$.

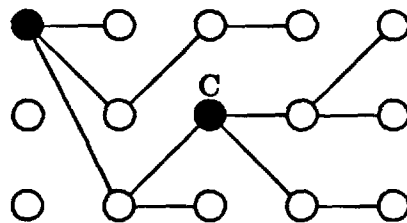


Figure 4.10: A convergence node of weight 2, the node $\{C\}$.

because it is the solution we are looking for, and it is the solution the trellis decoding algorithm will provide when operating on the overall speech sequence (the L layers). However the great motivation behind the concept of convergence node is to be able to process these “common tails” of depth $K \ll L$ as independent blocks. Can the TDA decode these blocks “optimally”? This is possible if carefully done. The first method, as described just above, applies the (unconstrained) TDA on $D = K + w$ layers and “outputs” the block of depth K , independently of other blocks. However the blocks are not really processed independently, since a given block of depth K needs the information present in the w layers after the block. The second method applies the (constrained) TDA on the block of depth K , independently of the other blocks, assuming an a priori knowledge of K and of the final (convergence) node of the block. The terms “unconstrained” and “constrained” will now be defined. First of all a block between two convergence nodes is really independent of past and future layers. Finding the overall best⁶ survivor is equivalent to finding each survivor in each block starting and ending in the two convergence nodes of the block, and concatenating them. This is true because the common tail of the N survivors emerging from any single convergence node can be discarded: the common tail has a single length that does not influence the minimization of the lengths over the upcoming layers (see figure 4.8). So now if we were to apply the trellis decoding algorithm on each of these individual, independent blocks, would the desired solution (the path $\{abcde\}$ in figure 4.8) be the “optimum path”? No, it would not necessarily be. Therefore the term “optimum path” is inappropriate here. The “optimum path”, i.e., the *shortest survivor* provided by the TDA (described so far) on the given block would not necessarily be the desired solution (like the path $\{abcde\}$ in figure 4.8). The solution path ending in the convergence node at time n , although being a survivor, would not necessarily be the shortest survivor at time n , but would only be the *survivor ending in the convergence node*. The existence of the convergence node was determined at time

⁶ “best” might mean the shortest or the one starting and ending in the right nodes.

$n + w$ and had no bearing on a fact such as ending the shortest survivor at time n . It just guaranteed that the N paths at time $n + w$, extended from the convergence node, would be the shortest. The decision essentially depended on the lengths of the extension branches (and, of course, on the lengths of the survivors from which they were extended). However, it would also generally be probable that the shortest survivor at time n would also be the solution path (the one ending in the convergence node), since the extension lengths would usually not dramatically alter the ordering of the survivor lengths at time $n + w$, especially if the survivor lengths at time n were significantly different.

As a summary, the true solution to the block-decoding problem is the optimum path provided by the *constrained* TDA applied on a block of layers of depth the convergence length. The term "constrained" means that the path search is constrained on the knowledge of the initial and final convergence nodes of the block. The expression "optimum path" means the survivor of depth K starting and ending in the corresponding convergence nodes, not necessarily the shortest one. The solution provided by the *unconstrained* TDA on the same block, would not necessarily be the true solution. However, the unconstrained TDA also provides the true solution — a block of depth K — when operating on $K + w$ layers — a block of depth $K + w$. We are now left with two modes of operation of the TDA:

- natural convergence and the unconstrained TDA: letting the unconstrained TDA run freely, one looks for spontaneous convergence nodes. Whenever a convergence node is found ⁷, the block solution is known to be the "common tail" between the previous and the present convergence node. This decoded block can be "output", and the TDA can be restarted from the current convergence node. Here the decoder delay and memory requirements are

⁷A maximum allowable convergence length can be used, at the risk of introducing errors.

kept to a minimum, and the optimality of the algorithm is preserved.

- forced convergence and the constrained TDA: if no natural convergence appears (or would have appeared for a convergence length greater than the maximum allowable convergence length) the constrained TDA should be used. Convergence nodes are forced on the survivors. This is also called "pegging" since nodes are pegged every K layers. The pegging period K might be constant or variable. Pegging might result in extraneous errors, unless the actual pegged nodes are known. Moreover, knowing that it takes a few layers for the TDA to stabilize, it is more robust to peg Q nodes at a time, instead of only one. However, pegging Q nodes at a time, instead of one, does not improve the degree of reliability of the TDA. It only improves the robustness (accuracy) of the decoded sequence (see figure 4.13).

The constrained TDA is expected to perform better than the unconstrained TDA, when it operates on *short segments* or *blocks* of speech, and processes the whole block at a time, independently of other blocks. The former algorithm might introduce local errors due to an approximate knowledge of the pegged nodes, but it operates under the right concept of optimality. The latter algorithm, even though it finds an error-free solution based on its own concept of optimality, is less preferred because it works under a partially correct concept of optimality, with respect to the block — this is especially true for the speech observation decoding. The previous TDA of figure 4.7 can be easily modified to include the search of convergence nodes (according to the partial backtracking algorithm described above) and the corresponding processing of blocks (this algorithm is the unconstrained TDA, with detection of convergence nodes).

In the case of speech decoding both modes of operation will be used; the choice of the mode is dictated by the type of decoding, the number of nodes N , and the most reliable information available. For the state-decoding problem, the

true model λ and the right expression for the likelihood are known, and the states cannot be observed. The number of nodes in each layer ($N = 64$) is reasonably small. Natural convergence will prevail. Therefore the unconstrained TDA should be used. For the observation-decoding problem, the true model λ is known, but the likelihood is uncertain. However the pegged observations are exactly known. The number of nodes ($N = 1024$) is close to what is generally considered the maximum limit under which the TDA can be used efficiently. Here, the constrained TDA should be used.

Let us finally introduce here a new concept: the global state/observation decoding. One of our main concerns has always been to use HMM's of speech to identify *global structures*. This was true in the way the model was formulated, in the way it was trained, and in the way the state decoding and the observation decoding were performed, independently. More globality can be reached, if the state and observation decoding are not considered as two independent problems, but as only one problem, with two interleaved processes, represented by the state and the observation sequences. A global decoding scheme should be seen as a "feedback decoding:" the optimum decoded state sequence should be the one which guarantees the optimum decoded observation sequence. In other words, the optimality of the state sequence is directly linked to the optimality of the observation sequence. Above, the notion of optimality was measured by the constrained or unconstrained log-likelihood. The notion directly emerged from the HMM itself. But how well does it fit the notion of optimality we are mainly concerned with: speech intelligibility? Outside from the log-likelihood, several other measures of intelligibility [13] could be used (the Itakura-Saito measure, to mention only one), to describe the quality of the decoded observation sequence. The feedback state/observation decoding could operate in several ways. Contrary to what was said before, the constrained TDA could be used for the state decoding. The choice of the initial and final states would be constrained on the optimality

of the corresponding decoded observation sequence. This would be seen as a *block feedback decoding*. Alternatively, decoded observations could be continuously fed back to the state decoder; it would directly affect, not only the initial and final states, but also the states in between. This would be seen as a *continuous feedback decoding*. The observation decoding would still be performed by the constrained TDA. The optimality (intelligibility) of the decoded observation sequence would be measured; the degree of optimality would “drive” the state decoder. For the observations, the constrained TDA could also be replaced with other schemes, such as the minimization of the LPC log-likelihood distance between actual and decoded observations.

4.4 Sub-optimum trellis decoding algorithms

The TDA described in section 4.2 is optimum, i.e., it finds the exact solution to the ML decoding problem (as long as the memory requirements are met). In this section we consider sub-optimum algorithms in the sense that they might find only a near-optimum solution. The goal is still to obtain the exact solution but it will not necessarily be reached. The probability of missing the exact solution will be, however, kept to a minimum. The motivation behind sub-optimum algorithms is increasing the algorithm speed, but also decreasing the memory needs. Two types of sub-optimum algorithms will be presented: the first algorithm will keep track of only the N_S best survivors instead of N ($N_S < N$); the second algorithm will include the features of the first algorithm, and also limit the number of possible survivor extensions to the N_E best extensions instead of N ($N_E < N$).

The first algorithm will perform a *backward pruning*: when decoding layer n it will look backwards at layer $n-1$ and consider only the N_S shortest survivors for extension, where $N_S = N - N_B$ ($N_B > 0$). N_B is the backward pruning factor; the corresponding pruning rate R_B will also be used: it is the percentage of survivors

discarded. Discarding a survivor ending in node x_{n-1} is equivalent (as far as the result is concerned) to temporarily setting $G(x_{n-1}, x_n) = 0$ at time n . The setting to 0 is conditioned on the survivor length compared to other survivor lengths.

The second algorithm will add to the first one a *forward pruning*: when considering the N_S survivors to be extended from layer $n - 1$ to layer n , the algorithm will look forward at layer n and retain only the N_E most probable nodes x_n , given Y_n , as possible children, where $N_E = N - N_F$ ($0 < N_F \leq N_B$)⁸. N_F is the forward pruning factor; the corresponding pruning rate R_F will also be used: it is the percentage of nodes discarded in layer n . Discarding a node x_n is equivalent (as far as the result is concerned) to having $F(x_n, Y_n) = 0$. Only the N_E most probable nodes x_n , given Y_n , will be considered plausible. This is also called “setting a *rank* constraint” on the matrix F .

4.4.1 Backward pruning

How can we implement the backward pruning process described in the preceding paragraph? In every layer n the algorithm should extend only the N_S shortest survivors. These N_S survivors have N_S distinct final nodes. Let us denote this set of nodes by $\tilde{\mathcal{A}} = \{1, 2, 3, \dots, N_S\}$. An easy way to comprehend the problem is to think of $\tilde{\mathcal{A}}$ as a set of pseudo-nodes, constituting a pseudo-trellis (or a meta-trellis). The pseudo-nodes represent a subset of the actual nodes $\mathcal{A} = \{1, 2, 3, \dots, N\}$. Moreover, the set $\tilde{\mathcal{A}}$ is “variable” with time, in the sense that the constant set of indices $\tilde{\mathcal{A}}$ actually represents different “real” nodes, from \mathcal{A} , for different time layers. To identify the relationship between $\tilde{\mathcal{A}}$ and \mathcal{A} , a node mapping function q_t should be defined:

⁸The right-hand inequality is necessary to be able to maintain the desired number of survivors $N_S \leq N_E$.

$$q_t : \begin{pmatrix} \tilde{A} & \longrightarrow & A \\ i & \longmapsto & q_t(i) \end{pmatrix}$$

The node mapping function q_t is of course variable with time. The time index t will very often be dropped. The function q_t maps a pseudo-node onto a real node: $q_t(i)$ is the final node of the i^{th} shortest survivor in layer t . The function q_t is therefore defined according to the ordering of the survivors, by increasing lengths, at time t . If we define $j\text{min}$ as a function extracting the j^{th} smallest element in a list, and if "I" is the same index operator as before, then:

$$q_t(j) = Ij\text{min}[L_t(i)]. \quad (4.9)$$

The backward pruning scheme can be directly implemented, just by writing the TDA for the pseudo-trellis, and relating the pseudo-nodes to the real nodes through the node mapping function. The corresponding pseudo-code is shown in figure 4.11.

The new algorithm reduces the computational complexity to the order NN_sL , instead of N^2L . The parent of only N_s pseudo-nodes per layer need to be stored, together with the index mapping function for each layer. It requires a number of storage locations of the order $2N_sL$, instead of NL . A memory saving will be achieved if $N_s < \frac{N}{2}$, i.e., if at least 50% of backward pruning is used. If $N_s \geq \frac{N}{2}$, the memory requirements of the two algorithms are the same (when storing the parent of each real node in each layer). Further memory savings might be gained by using other data structures than arrays: link lists, to represent the survivors, would only use N_sL storage locations.

Now, how optimum is the TDA with backward pruning? This will depend on the "behavior" of the survivors, and on the pruning rate. If the lengths of two survivors are close to one another, discarding one survivor effectively is difficult. If

```

D = 10;  R = 74;  T = 1
For j = 1, N                                     ;initialize
    If  $G(j) < 10^{-D}$  then
         $\hat{G}(j) = -10^{R-T}$ 
    Else
         $\hat{G}(j) = \log_{10}[G(j)]$ 
    End If
     $L(j) = -\hat{G}(j) - \hat{F}(j, Y_1)$ 
End For j
For t = 2, L                                     ;trellis depth loop
    For i = 1, NS
         $q(i) = \text{Imin}_k[L(k)]$                        ;backward node mapping
    End For i
    For j = 1, N
         $E_m = 10^R$ 
        For i = 1, NS
             $E(i) = L[q(i)] - \hat{G}(q(i), j)$            ;extend lengths
            If  $\min(E(i), E_m) = E(i)$  then
                 $p_t(j) = q(i)$                      ;find parent node
                 $E_m = E(i)$ 
            End If
        End For i
         $\tilde{L}(j) = E_m - \hat{F}(j, Y_t)$                  ;update lengths
    End For j
    For j = 1, N
         $L(j) = \tilde{L}(j)$ 
    End For j
End For t
 $X_L = \text{Imin}_i[L(i)]$ 
 $X = X_L$ 
For t = L, 2                                     ;backtracking
     $X_{t-1} = p_t(X)$ 
     $X = X_{t-1}$ 
End For t

```

Figure 4.11: Unconstrained ML trellis decoding algorithm with backward pruning.

the lengths of two survivors are very far apart, it is probably very safe to discard the longer survivor. The measure of closeness of survivor lengths can be statistically evaluated: by applying the optimum (no pruning) TDA on relevant data, one can determine how much discrepancy between two survivor lengths is needed, on the average, for one of the survivors to be dropped over the upcoming layers. A discrepancy threshold can be set. It would allow the TDA to set its own rate of pruning automatically, according to the lengths encountered. A variable pruning rate would usually be more efficient than a fixed pruning rate. As seen before, the decision of the TDA is more reliable as time increases. Usually the algorithm might start with small discrepancies between the survivor lengths (very small pruning rates should be used), and then proceed with increasing discrepancies (larger and larger pruning rates could be used). Note that setting the backward pruning rate to the highest possible level ($N_S = 1$) would be equivalent to building the solution, sequentially, as the concatenation of the shortest branches. It would not (necessarily) be the solution to the ML decoding problem, because, as explained before, it does not take into account the dependencies between the branch lengths.

4.4.2 Forward pruning

The forward pruning can be implemented in a way similar to the backward pruning. The algorithm is concerned with only the N_E most plausible nodes in the upcoming layer. We define the set of the plausible nodes by $\bar{\mathcal{A}} = \{1, 2, 3, \dots, N_E\}$. These pseudo-nodes generate a pseudo-trellis. The relationship between the pseudo-nodes and the actual nodes is represented by a forward mapping function:

$$r_t : \begin{pmatrix} \bar{\mathcal{A}} & \longrightarrow & \mathcal{A} \\ i & \longmapsto & r_t(i) \end{pmatrix}$$

where: $r_t(j) = I_j \min_i [-\hat{F}(i, Y_{t+1})]$.

$r_t(j)$ is the j^{th} most plausible node in layer $t + 1$. These are the nodes in layer $t + 1$, to which the survivors in layer t should be extended. We previously said that the forward pruning scheme was added on top of the backward pruning scheme. Actually two combinations are possible:

- the forced backward pruning described in section 4.4.1 is augmented by the forward pruning scheme with $N_S \leq N_E$. The algorithm of figure 4.11 is easily modified to act directly on the forward pseudo-trellis, through the forward mapping function.
- If a forced backward pruning is not used, then the forward pruning will generate its own *natural* backward pruning with $N_S = N_E$. Here the N_S survivors are not selected on the basis of their lengths, but on the basis of their final node. The backward pruning function is easily modified accordingly, just by setting $q_t(j) = r_t(j)$. The pseudo-code for this algorithm is shown in figure 4.12. The notation convention for $L(i)$ has been changed: instead of being the length of the survivor ending in the actual node i , $L(i)$ is now the length of the survivor ending in the pseudo-node i , i.e., in the actual node $r(i)$. This requires only N_E array elements.

The computational cost of the algorithm is of the order $N_E N_S L$. The memory cost is basically the same as the one of the algorithm with only backward pruning. One can choose to store the parent of each real node (some real nodes do not have a parent), or store the parent of each pseudo-node and the mapping function. In terms of optimality, the second algorithm (the one with forward and natural backward pruning) might be preferred, especially in the case of speech observation decoding. As mentioned earlier, the forward pruning is equivalent to setting a rank constraint on the matrix $F(i, k)$. In other words the algorithm sets a maximum allowable rank r_m , by selecting a given forward pruning rate. The existence and value of an actual maximum rank r_m (less than the number of nodes), can be statistically determined, using the training and state decoding phases. Once an observation sequence Y has been decoded in terms of its state sequence X , the

```

D = 10;  R = 74;  T = 1
For j = 1, N                                     ;initialize
    If  $G(j) < 10^{-D}$  then
         $\hat{G}(j) = -10^{R-T}$ 
    Else
         $\hat{G}(j) = \log_{10}[G(j)]$ 
    End If
     $r(j) = \text{Ijmin}_i[-\hat{F}(i, Y_1)]$ 
     $L(j) = -\hat{G}[r(j)] - \hat{F}(r(j), Y_1)$ 
End For j
For t = 2, L                                     ;trellis depth loop
    For i = 1, NS
         $q(i) = r(i)$                                      ;backward node mapping
    End For i
    For j = 1, NE
         $E_m = 10^R$ 
         $r(j) = \text{Ijmin}_i[-\hat{F}(i, Y_t)]$                      ;forward node mapping
        For i = 1, NS
             $E(i) = L(i) - \hat{G}[q(i), r(j)]$                  ;extend lengths
            If  $\min(E(i), E_m) = E(i)$  then
                 $p_i[r(j)] = q(i)$                          ;find parent node
                 $E_m = E(i)$ 
            End If
        End For i
         $\tilde{L}(j) = E_m - \hat{F}(r(j), Y_t)$                      ;update lengths
    End For j
    For j = 1, NE
         $L(j) = \tilde{L}(j)$ 
    End For j
End For t
 $X_L = r(\text{Imin}_i[L(i)])$ 
 $X = X_L$ 
For t = L, 2                                     ;backtracking
     $X_{t-1} = p_t(X)$ 
     $X = X_{t-1}$ 
End For t

```

Figure 4.12: Unconstrained ML trellis decoding algorithm with forward and natural backward pruning.

rank r_m of the least probable (plausible) observation per state can be determined. Then statistically, one would not expect an observation of rank greater than r_m , to be generated in any state⁹ (in other words, the observation has to be at least the r_m^{th} most probable observation in a state, to be generated by that state). Then setting a forward pruning to $N_E = r_m$ is sure to guarantee an optimum decoding (none of the actually plausible nodes will be discarded).

In practice, for the speech observation decoding, the trellis is composed of $N = 1024$ nodes. If the 64 states were clustering the observations uniformly, then only 16 observations would be plausible in each state. Therefore the TDA could be operated on a reduced trellis of only 16 pseudo-nodes, which would save a great deal of computations. As will be seen later, such a clean cut at 16 observations per state does not exist, and the actual maximum rank will have to be set higher than the optimistic 16 nodes. Another interesting feature of the algorithm with forward and natural backward pruning, is the possible automatic inclusion of pegged nodes through the forward mapping function.

The limiting case $N_E = 1$ is equivalent to finding the most probable observation per node. When the number of paths searched becomes very small ($N_S, N_E \rightarrow 1$), the TDA will make decoding errors. The TDA behaves then like a sequential decoding algorithm [35,20,64], except that only a forward move is allowed. If sequential decoding was to be used, backward moves should be allowed. The algorithm would move backwards, and follow another path, when it detects that the path it is currently following is wrong. The decision would be made on the comparison of the actual path length with an expected path length (a statistically meaningful threshold). Moreover, a biased log-likelihood function would be used: the biased log-likelihood function would decrease when the algorithm follows

⁹One could also define a maximum rank r_{mi} for each state i , especially when the states do not behave uniformly.

the right path, and increase when it follows an incorrect path. It would also be normalized to be able to compare paths of different depths. Outside the information contained in the nodes and the classical log-likelihood, extra-information in terms of heuristics would usually be necessary. This extra-information might or might not be included into the definition of the biased log-likelihood. Basically, a sequential decoding searches a tree, instead of a trellis, based on this information. The algorithm does not exactly minimize the log-likelihood, but extends a single path, as long as the biased log-likelihood decreases. It backs up and explores a different path when the biased log-likelihood starts increasing. The algorithm stops when the final node is reached. A reduction of the computational complexity is generally achieved, but the amount of computations is no longer constant with time. Such algorithms have been developed: the Fano algorithm [49] and the stack algorithm [3,20,63,70]. They are usually applied to a trellis beyond 1024 nodes per layer.

4.5 Constrained trellis decoding algorithm

The notion and necessity of a constrained algorithm was introduced in section 4.3. As seen before, it relates to the concept of forced convergence, and to the mechanism of node pegging. It allows not only a more efficient algorithm in terms of memory requirement, but also a more reliable decoding in terms of finite, limited predictability of speech. The accuracy of the unconstrained TDA decreases with time, past a given number of trellis layers: this is because the degree of predictability of speech decreases over long periods of time. A very long-term prediction, based on an initial state only, is difficult. The algorithm needs periodic or quasi-periodic information about the actual state of the system. Knowing the initial and final states of the system, the TDA can perform a more reliable estimation of the states in between, based on the minimization of the log-likelihood. The unconstrained TDA minimizes the log-likelihood over all possible paths in the trel-

lis. The constrained TDA will minimize the log-likelihood over all possible paths, starting and ending respectively in the initial and final nodes. These two nodes are the nodes to be pegged. They are assumed to be known with certainty (or at least with a high degree of certainty). As already mentioned, the constrained TDA will be most helpful for the speech observation decoding. The pegged nodes will be initial and final known observations. Given the state sequence, the initial and final observations — and therefore a preset pegging period — and an appropriate log-likelihood, the TDA will attempt to recover the observations in between the pegged observations. Outside from this information, additional heuristics (based on speech knowledge) might be necessary to synthesize intelligible speech. The initial and final nodes of the trellis will be respectively denoted by X_i and X_f . A schematic representation of the constrained TDA is shown in figure 4.13. The straight horizontal lines are actually a convenient representation of the path of the pegged nodes in the trellis — not necessarily straight in reality. The straightness is just meant to say that the actual path over this period of time is known, and the TDA need not be applied there. The pegging period is K . The number of pegged nodes is Q ($K = 6$, $Q = 3$ in figure 4.13). A sequence segment (a block), like the one inside the dashed box of figure 4.13, can be processed by the constrained TDA independently of other segments. The pseudo-code of the constrained TDA is easily derived from the pseudo-code of the unconstrained TDA: instead of doing the backtracking over the shortest survivor, it should be done over the survivor ending in the pegged node.

4.6 State and observation decoding for speech

4.6.1 State decoding

Given an observation sequence $Y(1:L)$ (denoted Y here for simplicity), the optimum state sequence $X(1:L)$ (denoted X) is the one which maximizes the likelihood.

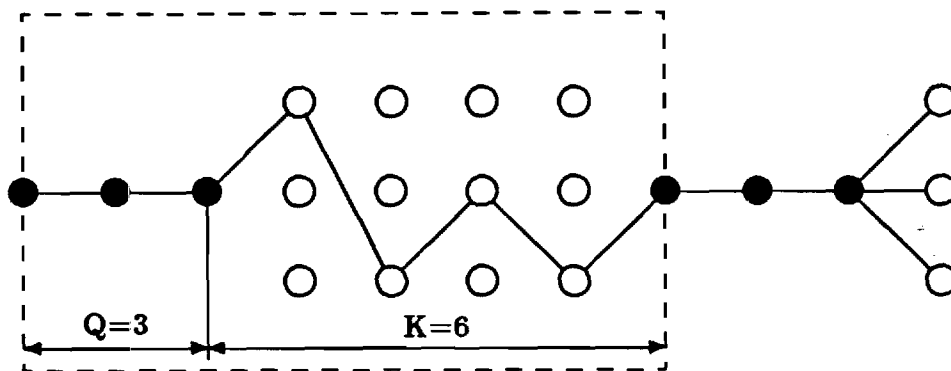


Figure 4.13: Operation of the constrained TDA.

The problem is to find X such that:

$$P(X, Y) = \max_x P(x, Y) = \max_x \Pr(Y/x) \Pr(x).$$

As explained earlier, the solution to this problem is given by the unconstrained TDA acting on the state-space trellis of the direct HMM λ , with detection of convergence nodes through partial backtracking. The states (nodes) of the trellis are also the speech states, and the trellis observations are the speech observations. The likelihood function is expressed by equation (4.1). The algorithms of this chapter directly apply to the function \mathcal{F} given in equation (4.2) with:

$$F(x_n, Y_n) = b_{x_n}(Y_n)$$

$$G(x_{n-1}, x_n) = a_{x_{n-1}x_n}.$$

4.6.2 Observation decoding

Several observation decoding schemes with different levels of optimality and complexity are possible. The intelligibility and quality of the synthesized speech is a function of the type of approach and algorithm used.

4.6.2.1 The most probable observation decoding

This scheme chooses the most probable observation in each of the states, i.e., given the known state sequence $X(1:L)$ (denoted X), it selects the observation sequence

$Y(1:L)$ (denoted Y) on the basis of:

$$\forall n = 1, L \quad b_{X_n}(Y_n) = \max_{y_n} b_{X_n}(y_n).$$

This is the simplest observation decoding scheme. It reduces the original codebook of M vectors to only s significant vectors (one in each state). The overall HMM coder/decoder can be thought of as a new vector quantizer with a reduced codebook of s vectors. To encode the speech — i.e., select the “best” match from the reduced codebook — the state sequence is detected by the TDA (instead of minimizing the Itakura distance between the speech vector and the vectors from the reduced codebook). At first sight this hybrid HMM-VQ speech encoding scheme might seem better than the classical VQ scheme, because it takes into account the long term dependencies across frames (instead of only a local distance minimization on a frame by frame basis). Unfortunately the HMM synthesized speech should at best be equal in quality to the VQ speech, since the HMM is only capable of modelling the frame dependencies which were present in the training corpus, i.e., that were encoded by the initial VQ representation based on the M -level codebook.

This most probable observation decoding scheme does not take full advantage of the capabilities of HMM's. There is a loss in observation resolution since the codebook is reduced from a size M to s . This decoding scheme does use some of the clustering information present in each of the states, as well as some of the language information encoded into the connections between states. Even though the decoding scheme uses only the local information of the current state X_n , this information was globally determined during the state decoding phase to satisfy the “connectionist” constraints of the state sequence based on the observed speech. However, it does not use all the connectivity properties of the HMM since observations are selected independently of one another based only on the value of the current state. A better decoding scheme would need to incorporate both state and observation dependencies by globally (and not locally) selecting an optimum

observation sequence Y based on the known state sequence X (in a way similar to the state decoding scheme).

4.6.2.2 The expected observation decoding

This decoding scheme is similar to the previous one, except that instead of using only one significant observation per state, it uses the M observations in the state, weighted according to their probabilities of occurrence. The decoded observation vector is given by:

$$\forall n = 1, L \quad \mathbf{O}_n = \sum_{k=1}^M b_{X_n}(k) \mathbf{O}_k,$$

where \mathbf{O}_k is the k^{th} LPC vector in the codebook. Actually, instead of directly averaging the LPC vectors (like prediction coefficients), the decoder averages the autocorrelation sequences. Here too the original codebook of size M is reduced to a codebook of size s , but the vectors of the reduced codebook do not belong anymore to the original codebook. The new spectral information encoded into the new vectors is a smoothed version of the spectral information described by the original vectors. Still this approach does not use all the connectivity properties of HMM's.

4.6.2.3 Maximum likelihood observation decoding

In order to take advantage as much as possible of most of the capabilities of HMM's, it seems intuitively reasonable to perform a maximum likelihood observation decoding, i.e., maximize the joint probability of the state and observation sequences. In other words, given a sequence $X(1:K)$ (denoted X), one would like to find the sequence $Y(1:K)$ (denoted Y) such that:

$$P(X, Y) = \max_y P(X, y) = \max_y \Pr(X/y) \Pr(y).$$

Unfortunately, formulated as it is, this approach is nothing more than the most probable observation decoding scheme since:

$$\max_y P(X, y) = \max_y \Pr(y/X) \Pr(X),$$

which is equivalent to finding:

$$\max_y \Pr(y/X) = \max_y \prod_{n=1}^K b_{X_n}(y_n),$$

i.e., finding:

$$\forall n = 1, K \quad b_{X_n}(Y_n) = \max_{y_n} b_{X_n}(y_n).$$

However, as explained earlier, a more proper approach is a maximum likelihood observation decoding constrained on the knowledge of the initial and final observations. Now the problem is to find:

$$P(X, Y) = \max_y P(X, y) = \max_y \Pr(X/y) \Pr(y) \quad \text{with} \quad y_1 = y_i, \quad y_K = y_f.$$

As shown earlier, the solution to this problem is given by the constrained TDA operating on the observation-space trellis of the adjoined HMM $\tilde{\lambda}$. The states (nodes) of the trellis are speech observations. The trellis observations are speech states. The expression (4.1) of the likelihood is still valid, but should be expressed in the terms of equations (2.31) and (2.32), i.e.,

$$P(X, y) = \prod_{n=1}^K \Pr(X_n/y_n) \Pr(y_n/y_{n-1}).$$

The algorithms of this chapter directly apply to the function \mathcal{F} with:

$$F(X_n, y_n) = \Pr(X_n/y_n) = \tilde{b}_{y_n}(X_n)$$

$$G(y_{n-1}, y_n) = \Pr(y_n/y_{n-1}) = \tilde{a}_{y_{n-1}, y_n},$$

where these probabilities are characteristic of the adjoined HMM (see section 2.3). The pegging period K defined earlier (section 4.3) as the number of trellis layers between the initial (y_i) and final (y_f) pegged observations, should be short enough: when K becomes large (roughly, greater than 12 layers), the constrained ML-TDA

becomes similar to the unconstrained TDA, i.e., to the most probable observation decoding scheme. As will be seen later, the constrained TDA for observation decoding will produce satisfactory speech intelligibility. However, even though this observation decoding scheme is better than the other schemes described previously, it is still not optimum¹⁰.

4.6.2.4 The observation decoding as an inverse problem

Ideally one would like the observation decoding process to be the inverse of the state decoding process. The state decoding performs a data compression. The observation decoding should perform an inverse data expansion. So far state and observation decoding were summarized as follows:

- state decoding:

Y given, find X s.t. $P(X, Y) = \max_x P(x, Y)$

- observation decoding:

X given, find Y s.t. $P(X, Y) = \max_y P(X, y)$ with $y_1 = y_i$, $y_L = y_f$.

The likelihood $P(X, Y)$ can be thought of as a function of two variables (the sequences X and Y). Initially for a fixed original Y_o the state decoding finds the optimum X . Now for the same X let us consider all the different possible sequences y 's and compute $P(X, y)$. The maximum of this function over y is reached at $y = \hat{Y}$, i.e.,

$$\max_y P(X, y) = P(X, \hat{Y}).$$

Obviously \hat{Y} is not necessarily equal to Y_o , that is, the ML observation decoding is not the inverse of the state decoding (and therefore is not optimum in terms of speech quality)¹¹. To summarize, the operation of the ML state and observation

¹⁰By using slightly different likelihood functions, and/or a sequential decoding algorithm (such as the stack algorithm) one could possibly improve the quality of the synthesised speech a little more. However, this would still rely on the same basic approach.

¹¹In general, however, \hat{Y} will be a "good" approximation of Y_o .

decodings is shown in figure 4.14:

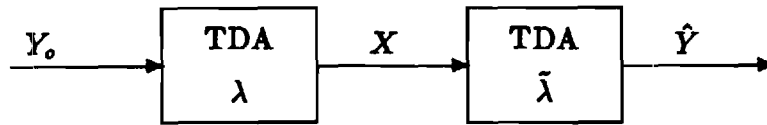


Figure 4.14: Global operation of the state and observation decodings.

To implement a better observation-decoding scheme one can choose between two strategies. One strategy would be to base the approach on the adjoined model $\tilde{\lambda}$ and use a better non-ML decoding algorithm. Another strategy would keep a ML decoding algorithm but based on a different model η with associated $P_\eta(X, Y)$. We will adopt the latter formulation and define an *inverse* HMM λ^{-1} . The definition, existence, and derivation of λ^{-1} should be established through a criterion of optimality of the synthesized speech, according to the diagram of figure 4.15:

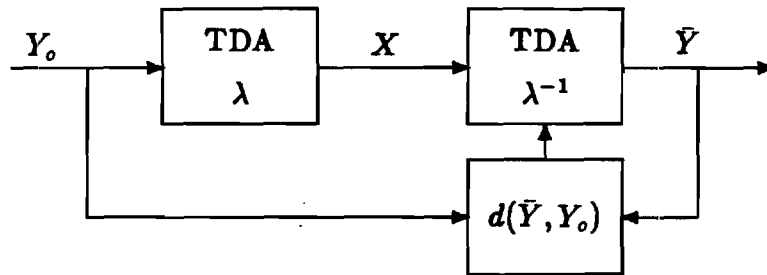


Figure 4.15: Principle of the estimation of the inverse HMM λ^{-1} .

The inverse model λ^{-1} should be estimated to minimize the distance between the decoded and original speech $d(\tilde{Y}, Y_o)$. In the ML observation decoding approach the adjoined model $\tilde{\lambda}$ was actually used as an approximation of λ^{-1} . Let us denote the respective likelihood functions of $\tilde{\lambda}$ and λ^{-1} by $P_{\tilde{\lambda}}$ and $P_{\lambda^{-1}}$. Note that previously we used the notation $P(X, Y)$ to mean $P(X, Y/\lambda)$. Now the notation $P_\lambda(X, Y)$ will be used to denote $P(X, Y/\lambda)$. This likelihood, as seen before, can be expressed in two different ways: $P_\lambda = \Pr(Y/X) \Pr(X)$ is directly expressed in terms of the parameters of the model λ , or $P_\lambda = \Pr(X/Y) \Pr(Y)$ is directly

expressed in terms of the parameters of the model $\tilde{\lambda}$. The latter expression will also be denoted $P_{\tilde{\lambda}}$. The model $\tilde{\lambda}$ is a convenient way of expressing the likelihood for the purpose of observation decoding. It is not a “new model” because it contains the same information as the original model λ , from which it was derived — $P_{\lambda}(X, Y) = P_{\tilde{\lambda}}(X, Y)$. On the contrary, λ^{-1} is a new model. $P_{\lambda^{-1}}(X, Y)$ is also directly expressed in the form $\Pr(X/Y) \Pr(Y)$, as in the case of $\tilde{\lambda}$, but now $P_{\lambda^{-1}}(X, Y) \neq P_{\lambda}(X, Y)$. The general behavior of the functions $P_{\tilde{\lambda}}$ and $P_{\lambda^{-1}}$ should be expected to look like the graphs of figure 4.16:

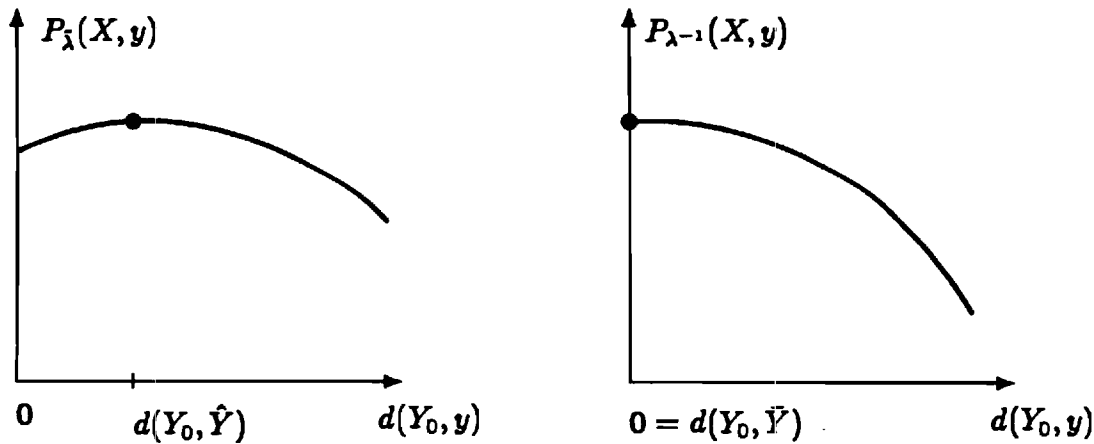


Figure 4.16: Behavior of the ML observation decoding based on $\tilde{\lambda}$ and λ^{-1} .

The maximization of $P_{\tilde{\lambda}}(X, y)$ over y leads to a solution \hat{Y} which does not minimize $d(\hat{Y}, Y_0)$. On the contrary, the maximization of $P_{\lambda^{-1}}(X, y)$ over y leads to a solution \bar{Y} which minimizes $d(\bar{Y}, Y_0)$. One should ideally use the optimum approach based on λ^{-1} . Section 4.6.3 will describe how λ^{-1} can be estimated.

Regarding the choice of the distance measure $d(\bar{Y}, Y_0)$, one could start with a simple Hamming distance since \bar{Y} and Y_0 are drawn from the same alphabet. But merely counting the number of times \bar{Y} and Y_0 differ would probably not be sufficient. Suppose $Y^{(1)}$ and $Y^{(2)}$ are two decoded sequences with $d(Y^{(1)}, Y_0) = 1$

and $d(Y^{(2)}, Y_o) = 2$. In other words, $Y^{(1)}$ differs from Y_o once, and $Y^{(2)}$ twice. But still $Y^{(2)}$ could be “closer” to Y_o than $Y^{(1)}$ is, in terms, for example, of spectral distance. Therefore the next step would be to select a more sophisticated distance measure, such as the Itakura distance between LPC vectors indexed by \bar{Y} and Y_o .

Finally schemes other than the TDA could be used. A decoding diagram — previously referred to as a global state/observation feedback decoding — with a decoding function H is shown in figure 4.17. This model could be used to estimate not only λ^{-1} but also λ .

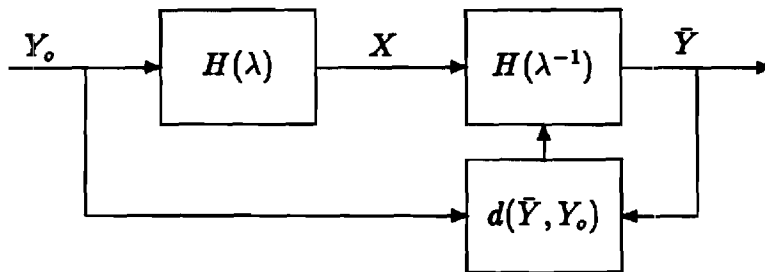


Figure 4.17: Global feedback estimation of λ^{-1} , and possibly λ .

4.6.2.5 Exact reconstruction

If the value of the likelihood is transmitted (or specified) together with the state sequence X , exact reconstruction of the observation sequence is possible most of the time: in theory one could go through all possible observation sequences and derive the corresponding state sequence for each of them. If this decoded state sequence is equal to the exact state sequence X , then the observation sequence under consideration is a good candidate for the reconstructed observation sequence. Among all the good candidates, only the observation sequence with the correct likelihood would be retained as the correct reconstructed observation sequence. This scheme would almost always find the exact observation sequence, except pos-

sibly in the case of likelihood ties¹².

Unfortunately, this observation decoding scheme for which an exhaustive search of the observation space is performed, is not practical: there are too many possible observation sequences (M^L for sequences of depth L). However, it is easy to write a fairly efficient algorithm that would perform this decoding without going through an exhaustive search. For the sake of clarity, let us introduce the concepts of *plausible* and *admissible* observation sequences with respect to a given state sequence $X(1:L)$. We say that an observation sequence $Y(1:L)$ is plausible with respect to the state sequence $X(1:L)$ if $X(1:L)$ can generate $Y(1:L)$. If the output probability matrix B does not have any zero entries, then there are M^L plausible observation sequences with respect to the given state sequence. We say that an observation sequence $Y(1:L)$ is admissible with respect to the state sequence $X(1:L)$ if the maximum likelihood state decoding applied to $Y(1:L)$ produces $X(1:L)$ (Y was called a "good candidate" just above). Note that all plausible sequences Y 's are not necessarily admissible. However, an admissible sequence has to be a plausible sequence. Therefore the set of admissible sequences with respect to a given state sequence, S_A , is a subset of the set of plausible sequences with respect to the same state sequence, S_P .

It is possible to write a fairly efficient recursive algorithm to generate the set of admissible sequences S_A . It is based on a few simple properties of admissible sequences. Note that the exact reconstructed observation sequence has to be admissible. If $Y(1:n)$ is admissible with respect to $X(1:n)$, then $\{Y(1:n-1), \hat{y}_n\}$ is admissible with respect to $X(1:n)$ for any \hat{y}_n (this is easily seen from the operation of the state decoding TDA). The same property is valid for $\{\hat{y}_1, Y(2:n)\}$ if the state

¹²In the case of infinite precision arithmetic (or, as a good approximation, floating point arithmetic), likelihood ties would be very highly improbable for our model λ . However, if the likelihood was quantized, likelihood ties would occur.

decoding TDA operated with a fixed known initial state. Therefore the same result applies to the sequences $\{\hat{y}_1, Y(2:n-1), \hat{y}_n\}$. If $Y(1:n-1)$ is admissible with respect to $X(1:n-1)$ and if $\{Y(1:n-1), \hat{y}_n\}$ is not admissible with respect to $X(1:n)$, then any sequence $\{Y(1:n-1), \hat{y}_n, Y(n+1:L)\}$ will not be admissible with respect to $X(1:L)$. Therefore all these sequences need not be searched for. If $Y(1:n-1)$ is admissible with respect to $X(1:n-1)$, then only one branch extension has to be computed in the state-space trellis at layer n to detect the non-admissible sequence $\{Y(1:n-1), \hat{y}_n\}$ with respect to $X(1:n)$ (by checking that the parent of X_n is not X_{n-1} for this \hat{y}_n). If $\{Y(1:n-1), \hat{y}_n\}$ is not admissible, then the algorithm checks the next possible \hat{y}_n in the same layer. If, in a given layer, all the possible observations have been exhausted, then the state decoding TDA moves one layer backward. If $\{Y(1:n-1), \hat{y}_n\}$ is admissible, the state decoding TDA moves one layer forward. The algorithm stops when the TDA moves back to the last observation of the first layer.

4.6.3 Estimation of the inverse HMM λ^{-1}

The principle of the estimation of λ^{-1} was described in section 4.6.2.4. The basic idea was to find the model η which minimizes the distance $d(Y_o, \bar{Y})$ between the original and reconstructed speech. Let us now denote the original observation sequence by Y (for $Y(1:L)$), and the corresponding state sequence by X . Let y be a possible decoded observation sequence. In this section, instead of dealing with a distance measure d , we will introduce a similarity measure $S(y, Y)$ with the following properties:

- the similarity measure should be additive: $S(y, Y) = \sum_{n=1}^L s(y_n, Y_n)$, or multiplicative: $S(y, Y) = \prod_{n=1}^L s(y_n, Y_n)$.
- $\forall k = 1, M \quad \forall \ell = 1, M \quad 0 \leq s(k, \ell) \leq 1$
- $\forall k = 1, M \quad \sum_{\ell=1}^M s(k, \ell) = 1$
- the "closer" the observations k and ℓ , the larger the measure $s(k, \ell)$.

Therefore the similarity measure behaves very much like a probability, and one can define a stochastic similarity matrix:

$$\mathbf{S} = (s_{k\ell})_{k=1,M;\ell=1,M} \quad \text{with} \quad s_{k\ell} = s(k, \ell).$$

Two examples of similarity measures are given below:

- the Kronecker (additive or multiplicative) similarity measure¹³:

$$s(k, \ell) = \delta(k, \ell) = \begin{cases} 1 & \text{if } k = \ell \\ 0 & \text{if } k \neq \ell \end{cases} \quad (4.10)$$

- additive: $S(y, Y) = \text{number of times } y \text{ and } Y \text{ agree}$
- multiplicative: $S(y, Y) = 1$ if $y = Y$, 0 otherwise.

- the Itakura (multiplicative) similarity measure:

$$s(k, \ell) = \frac{e^{d(k, \ell)}}{\sum_{m=1}^M e^{d(k, m)}}, \quad (4.11)$$

where $d(k, \ell) \geq 0$ is the classical Itakura distance measure (or log-likelihood ratio).

The estimation problem is now equivalent to maximizing, over the model η , the overall similarity measure between the original speech Y and all the possible decoded speech sequences y 's. If we denote the overall similarity measure by $S_\eta(X, Y)$ then:

$$S_\eta(X, Y) = \sum_y P_\eta(X, y) S(y, Y) \quad \text{i.e.,} \quad (4.12)$$

$$S_\eta(X, Y) = \sum_y S(y, Y) \prod_{n=1}^L \text{Pr}_\eta(y_n/y_{n-1}) \text{Pr}_\eta(X_n/y_n). \quad (4.13)$$

The problem is to find λ^{-1} such that:

$$S_{\lambda^{-1}}(X, Y) = \max_\eta S_\eta(X, Y). \quad (4.14)$$

¹³It could also be called the modified Hamming similarity measure.

Except for the common factor $S(y, Y)$ present in equation (4.13), this problem is identical to the estimation problem for the parameters of a classical hidden Markov model. Does the FBA still apply to the likelihood function defined by (4.13)? If yes, what are the new reestimation formulas? Can they be evaluated recursively? If the FBA does not apply we will have to rely on classical numerical optimization techniques.

Table 4.3: Analysis-Synthesis HMM's of Speech: summary.

Sub-Optimum Maximum Likelihood Approach	
Phase	Procedure
training (estimate λ)	$\sum_x P_\lambda(x, Y) = \max_\eta \sum_x P_\eta(x, Y)$
state decoding (estimate X)	$P_\lambda(X, Y) = \max_x P_\lambda(x, Y)$
observation decoding (estimate Y)	$P_\lambda(X, \hat{Y}) = \max_y P_\lambda(X, y)$ with $\hat{Y}_1 = Y_i, \hat{Y}_K = Y_f$
Optimum Maximum Likelihood Approach	
Phase	Procedure
training (estimate λ)	$\sum_x P_\lambda(x, Y) = \max_\eta \sum_x P_\eta(x, Y)$
state decoding (estimate X)	$P_\lambda(X, Y) = \max_x P_\lambda(x, Y)$
training (estimate λ^{-1})	$\sum_y S(y, Y) P_{\lambda^{-1}}(X, y) = \max_{\eta^{-1}} \sum_y S(y, Y) P_{\eta^{-1}}(X, y)$
observation decoding (estimate Y)	$P_{\lambda^{-1}}(X, \bar{Y}) = \max_y P_{\lambda^{-1}}(X, y)$ (with or without $\bar{Y}_1 = Y_i, \bar{Y}_K = Y_f$)

CHAPTER 5

HMM's and speech coding

HMM's can be used to reduce the amount of information necessary to encode the short term speech spectrum. The point of reference will be the classical 1024-level vector quantizer, which can encode the speech spectral information with 10 bits/frame. A speech coder trying to achieve a bit rate lower than this 10 bits/frame will be called a very-low-bit-rate speech coder¹. We will describe three types of very-low-bit-rate speech coders: a coder simply based on a (non-hidden) Markov chain of the VQ vectors called the *Markov-VQ*, and two coders based on the speech HMM called the *partitioned HMM-VQ* and the *HMM-VQ*.

5.1 The Markov-VQ

A simple Markov chain can be used to directly model the transitions in a sequence of LPC vectors drawn from a VQ codebook of size $M = 2^B$. This is equivalent to building an HMM with M states and a diagonal output probability matrix B . States and observations are the same thing: vectors from the VQ codebook. The Markov chain parameters can be directly estimated by frequency counts from a speech training sequence of LPC vectors. The original bit rate of B bits/frame can be reduced² to H_A bits/frame, where H_A is the entropy of the Markov chain (or

¹In this research we are not concerned with the encoding of the speech excitation source.

²For example, by using Huffman coding.

equivalently of the transition matrix). Table 6.3 in chapter 6 shows experimental entropies (and therefore bit rates) for vector quantizers with different codebook sizes.

5.2 The partitioned HMM-VQ

This coder, instead of transmitting only the sequence of observations $\{Y_t\}$, transmits both the state sequence $\{X_t\}$ and the observation sequence $\{Y_t\}$, and by doing so, allows a bit rate reduction³. This coder partitions the original codebook of size M into s sub-codebooks (or classes) of size M , indexed by the states⁴. Given a class (i.e., a state, or a sub-codebook), it takes much less information to encode the observations in this class (simply because in each class a small number of observations will be very probable, and the rest of the observations will be highly improbable). This coder uses a bit rate of $H_A + H_B$ bits/frame (this bit rate will be seen to be roughly 8 bits/frame — instead of 10 — in the next chapter). The coder transmits the exact observation sequence, therefore no error is made, and no reconstruction is necessary. A simple coding scheme is to generate s different Huffman codes (one code per class). In each class each observation is then given a Huffman codeword according to its probability of occurrence in that class.

5.3 The HMM-VQ

To lower the bit rate below $H_A + H_B$ bits/frame, another coding scheme is necessary. The HMM-VQ coder transmits only the state sequence $\{X_t\}$, and, for some of the decoding schemes, the actual observation at every pegging period K . Since

³A paradox, isn't it!

⁴If improbable but plausible errors were allowed, the size of the sub-codebooks could be reduced, for example to 2^{H_B} vectors — the 2^{H_B} most probable vectors in the given class. Notice that it is not a partition in the mathematical sense of the term because the classes are not necessarily disjointed.

the observation sequence is not transmitted, it has to be reconstructed between two pegged observations. The various reconstruction schemes were described in chapter 4. The bit rate of the coder is⁵:

$$B(K) = \frac{(K-1)H_A + H_B}{K-1}. \quad (5.1)$$

The minimum achievable bit rate is H_A (the state entropy) reached for K very large. For K very small (actually $K = 2$) this coder is equivalent to the partitioned HMM-VQ with a bit rate $H_A + H_B$. In chapter 6 we will give the experimental entropy results, and discuss the speech intelligibility as a function of the bit rate. Tables 6.4 and 6.5 in chapter 6 show the coder bit rate for different pegging periods K . Figure 5.1 shows the bit rate range (shaded area) as a function of the structuring capability of the HMM (the top curve is for an HMM showing no structure at all, the bottom curve is for our experimental HMM showing a significant structure). Numerical values for the bottom curve are shown in table 6.4. The intersection of the curve $B(K)$ with a constant level H ($H > H_A$) occurs at:

$$K_m = 1 + \frac{H_B}{H - H_A}. \quad (5.2)$$

K_m is the smallest pegging period necessary to achieve a bit rate lower than H .

5

$$B(K) = \lim_{n \rightarrow \infty} \frac{[1 + n(K-1)]H_A + (1+n)H_B}{1 + (K-1)n} = \frac{(K-1)H_A + H_B}{K-1}$$

$$\lim_{K \rightarrow 2} B(K) = H_A + H_B \quad \lim_{K \rightarrow \infty} B(K) = H_A$$

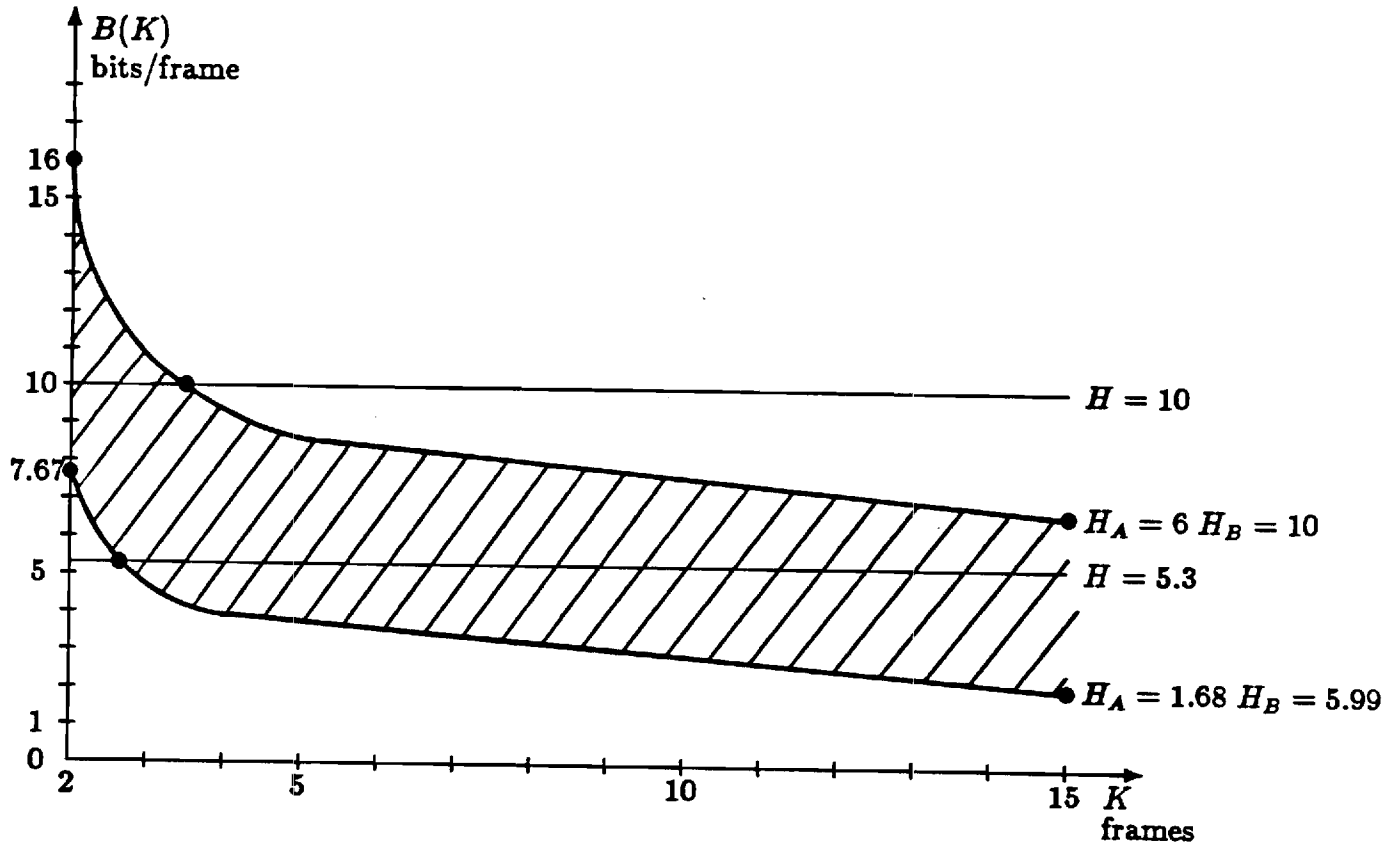


Figure 5.1: Range of the HMM-VQ bit rate as a function of the pegging period.

CHAPTER 6

Experimental results

An HMM with 64 states and 1024 observations was trained on actual speech. A database of continuous speech from one male American speaker was generated. The speech was 15 minutes long, and read from a script¹. The continuous speech was sampled at 8kHz, and quantized with 12 bits. It represented 60,000 frames (15ms frames, 120 samples/frame). A pitch file was generated using a semi-automatic pitch detector. A classical 10th order LPC analysis was performed (autocorrelation method). A 1024-level VQ codebook was generated with the classical algorithm [90,30] (K-means algorithm, gain normalized Itakura-Saito distance measure). The 60,000 frames of speech were vector quantized. The synthetic speech produced by the pitch excited LPC-VQ vocoder was generated. A 64-state, 1024-observation HMM was trained according to the method of chapter 3. State and observation decoding, based on this model, was carried out according to the methods of chapter 4. The following sections summarize the experimental results.

¹To estimate the $64^2 = 4096$ parameters of A , one would need 40,960 frames, on a basis of 10 occurrences of each parameter in the training data. To estimate the $64 \times 100 = 6,400$ significant parameters of B — assuming that roughly 100 observations will be significant in each state, — one would need 64,000 frames, on a basis of 10 occurrences per parameter. Therefore 15mn of speech (60,000 frames) is a minimum requirement in terms of training data size.

6.1 Training

Outside from the mathematical optimization described in chapter 3, there was also a need for optimizing the resources of the computer available to us². In that case, optimization was always a compromise between accuracy, storage, and speed. The optimization process is summarized in table 6.1. The optimized FBA was capable of one iteration on the 15 minutes of speech in 1 hour and 42 minutes of CPU time. Performing the whole training (99 iterations of the FBA) took one week on a single-user Data General MV/10000, and more than two months on the same computer used as a multi-user system³. The initial estimate will be called the model λ_1 , the estimate after one iteration of the FBA will be called the model λ_2 , etc.

In the FBA with partial scaling we used the values $m_a = m_b = 10^{-11}$ and $\beta_R = 2 \times 10^{-24}$. The entries of the initial estimate λ_1 were selected by a random number generator, and normalized so that the entries in each row sum to 1 (see figures 6.3 and 6.7 for the initial estimates of π_1 and A respectively). Table 6.2 shows the evolution of the log-likelihood and entropies of the model as a function of the iteration number. The corresponding graphs are shown in figures 6.1 and 6.2. The first iteration of the FBA provided a significant improvement of the initial estimates, as shown by the sharp decrease in the log-likelihood \mathcal{L} , and the output entropy H_B after the first iteration, compared to the slowly decreasing plateau thereafter. Most of the model improvement was, however, achieved between iteration 20 and 40. At iteration 100 the training process was certainly close to the

²A Data General MV/10000 with up to 15 Mbytes of virtual memory, or a Cyber 990, faster in terms of computations, but on the whole, slower, because of limited virtual memory available — 2 Mbytes at most.

³In the latter case, the factor between real time and CPU time was usually greater than 10.

convergence values. The likelihood was greatly improved by a factor:

$$\frac{P(X, Y/\lambda_{100})}{P(X, Y/\lambda_1)} \approx 10^{4.4 \times 10^4}.$$

The initial estimate λ_1 was random and fully connected, showing no sign of any significant structure. The corresponding entropies were $H_A(1) = 5.7$, and $H_B(1) = 9.7$. The model λ_{100} displayed a great deal of structure, and lowered the entropies significantly to $H_A(100) = 1.8$, and $H_B(100) = 6.1$. This means that, according to the model λ_{100} , roughly 4 transitions are plausible from any given state at any given time, and roughly 1 observation among 64 is likely to be produced from each state. Moreover, the graph of the entropies shows that a training based on a maximum likelihood approach (the FBA) leads also to a minimum entropy solution. The natural evolution of the model from the fully connected initial estimate λ_1 did not lead to a model λ_{100} with a left-to-right structure⁴, as seen for example from figures 6.10 and 6.11.

The evolution (as well as the rate of convergence) of the initial distribution of the states, as a function of the iteration number, is shown in figure 6.3. This distribution converges to the first state of the training sequence: state 23 in this case, a silence/noise state. Figure 6.4 shows the evolution of the steady-state distribution of the states⁵. Entries histograms for the transition matrix A and the output probability matrix B are shown in figures 6.5 and 6.6. It shows that the model performed a significant structuring and clustering by allowing only a small number of state transition probabilities, and a small number of observation probabilities to have a significant value. The emergence and evolution of the structure of the transition matrix is shown in figures 6.7, 6.8, 6.9, and 6.10. The entries

⁴However, to be conclusive, we would have to show that no state permutation could rearrange the model λ_{100} into a left-to-right structure.

⁵This distribution was directly derived from the model λ with equations (2.15), and not from a frequency count on the decoded state sequence.

of the transition matrix of figure 6.10 greater than 0.1 are shown in figure 6.11⁶. The strong diagonal structure, with a few extra significant transitions on the side, means a low entropy H_A , and a strong structuring capability of the model.

Table 6.1: The FBA: a long optimization process.

Algorithm type	CPU time†
original maximization problem	infeasible
Forward Backward Algorithm (FBA)	theoretically feasible practically infeasible
scaled FBA	CPU > 30 hours
partially optimized, scaled FBA	20 h < CPU < 30 h
optimized FBA with partial scaling	CPU < 20 h
optimized, log-less FBA with partial scaling	CPU = 1 hour 42 minutes

†For the algorithm operating on the 15min of speech on a Data General MV/10000.

⁶Figure 6.11 is the horizontal section of figure 6.10 by a plane at height 0.1.

Table 6.2: Log-likelihood and entropies as a function of the iteration number.

Model number	State entropy H_A	Observation entropy H_B	Log-likelihood $\mathcal{L} \times 10^{-5}$
1	5.7218	9.7220	1.80388
2	5.7216	9.5959	1.78224
3	5.7212	9.5950	1.78222
4	5.7207	9.5938	1.78218
5	5.7200	9.5921	1.78215
6	5.7191	9.5901	1.78209
7	5.7179	9.5875	1.78202
8	5.7165	9.5843	1.78195
9	5.7147	9.5804	1.78188
10	5.7126	9.5757	1.78180
11	5.7099	9.5699	1.78170
12	5.7066	9.5627	1.78159
13	5.7024	9.5536	1.78141
14	5.6967	9.5417	1.78118
15	5.6888	9.5255	1.78084
16	5.6765	9.5014	1.78027
17	5.6523	9.4598	1.77897
18	5.5834	9.3710	1.77530
19	5.3794	9.1758	1.76215
20	5.0478	8.8845	1.73229
21	4.5506	8.4795	1.68971
22	3.8080	7.9733	1.62502
23	3.2449	7.5833	1.55898
24	2.9500	7.3234	1.51792
25	2.7823	7.1321	1.49297

Table 6.2: Log-likelihood and entropies as a function of the iteration number
(continued).

Model number	State entropy H_A	Observation entropy H_B	Log-likelihood $\mathcal{L} \times 10^{-5}$
26	2.6681	6.9831	1.47532
27	2.5825	6.8614	1.46124
28	2.5141	6.7609	1.44999
29	2.4547	6.6761	1.44038
30	2.3978	6.6018	1.43189
31	2.3429	6.5371	1.42439
32	2.2895	6.4811	1.41772
33	2.2381	6.4342	1.41183
34	2.1909	6.3965	1.40650
35	2.1501	6.3662	1.40207
36	2.1158	6.3411	1.39837
37	2.0863	6.3186	1.39512
38	2.0610	6.2981	1.39209
39	2.0387	6.2789	1.38942
40	2.0190	6.2611	1.38705
41	2.0014	6.2448	1.38500
42	1.9856	6.2306	1.38313
43	1.9719	6.2187	1.38154
44	1.9598	6.2084	1.38012
45	1.9495	6.1991	1.37888
46	1.9411	6.1905	1.37773
47	1.9340	6.1829	1.37675
48	1.9278	6.1761	1.37590
49	1.9226	6.1700	1.37501
50	1.9181	6.1644	1.37437

Table 6.2: Log-likelihood and entropies as a function of the iteration number (continued).

Model number	State entropy H_A	Observation entropy H_B	Log-likelihood $\mathcal{L} \times 10^{-5}$
51	1.9141	6.1591	1.37368
52	1.9107	6.1539	1.37302
53	1.9078	6.1489	1.37245
54	1.9053	6.1441	1.37192
55	1.9029	6.1398	1.37145
56	1.9006	6.1357	1.37096
57	1.8981	6.1319	1.37049
58	1.8958	6.1284	1.37010
59	1.8937	6.1250	1.36973
60	1.8917	6.1218	1.36939
61	1.8900	6.1188	1.36904
62	1.8882	6.1161	1.36871
63	1.8863	6.1138	1.36840
64	1.8845	6.1115	1.36807
65	1.8827	6.1094	1.36780
66	1.8810	6.1074	1.36757
67	1.8792	6.1055	1.36734
68	1.8773	6.1035	1.36709
69	1.8753	6.1015	1.36684
70	1.8735	6.0996	1.36664
71	1.8720	6.0980	1.36640
72	1.8707	6.0965	1.36621
73	1.8695	6.0951	1.36604
74	1.8681	6.0937	1.36592
75	1.8667	6.0924	1.36579

Table 6.2: Log-likelihood and entropies as a function of the iteration number (continued).

Model number	State entropy H_A	Observation entropy H_B	Log-likelihood $\mathcal{L} \times 10^{-5}$
76	1.8654	6.0912	1.36563
77	1.8641	6.0900	1.36544
78	1.8630	6.0888	1.36528
79	1.8617	6.0877	1.36512
80	1.8605	6.0867	1.36497
81	1.8590	6.0860	1.36484
82	1.8575	6.0853	1.36472
83	1.8561	6.0848	1.36457
84	1.8548	6.0841	1.36443
85	1.8537	6.0834	1.36431
86	1.8524	6.0826	1.36420
87	1.8512	6.0817	1.36409
88	1.8502	6.0808	1.36392
89	1.8493	6.0800	1.36383
90	1.8485	6.0792	1.36373
91	1.8476	6.0784	1.36365
92	1.8467	6.0777	1.36353
93	1.8457	6.0771	1.36342
94	1.8447	6.0764	1.36333
95	1.8435	6.0758	1.36324
96	1.8424	6.0751	1.36313
97	1.8414	6.0745	1.36305
98	1.8406	6.0740	1.36297
99	1.8396	6.0735	1.36292
100	1.8386	6.0732	1.36284

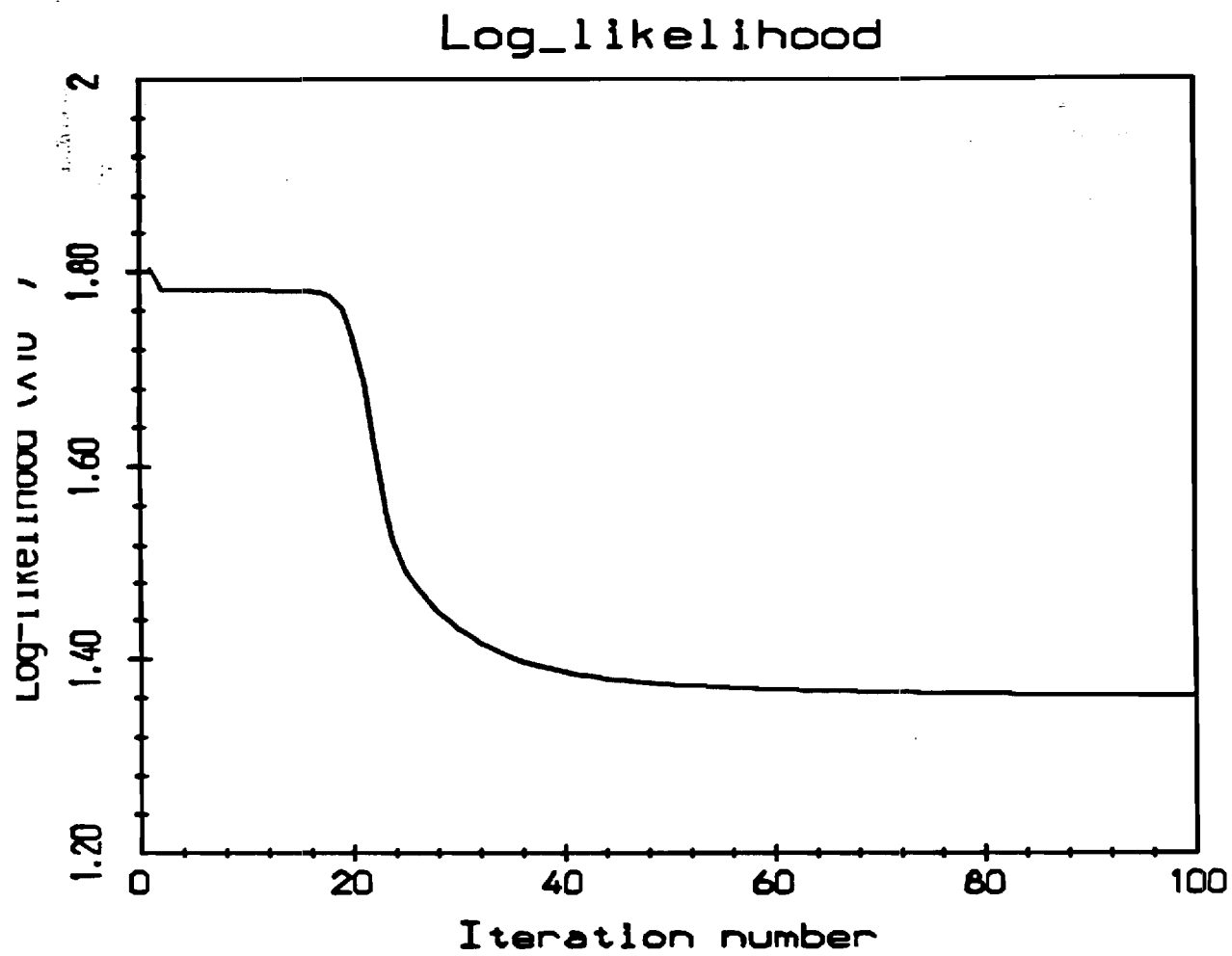


Figure 6.1: Evolution of the log-likelihood.

Entropies

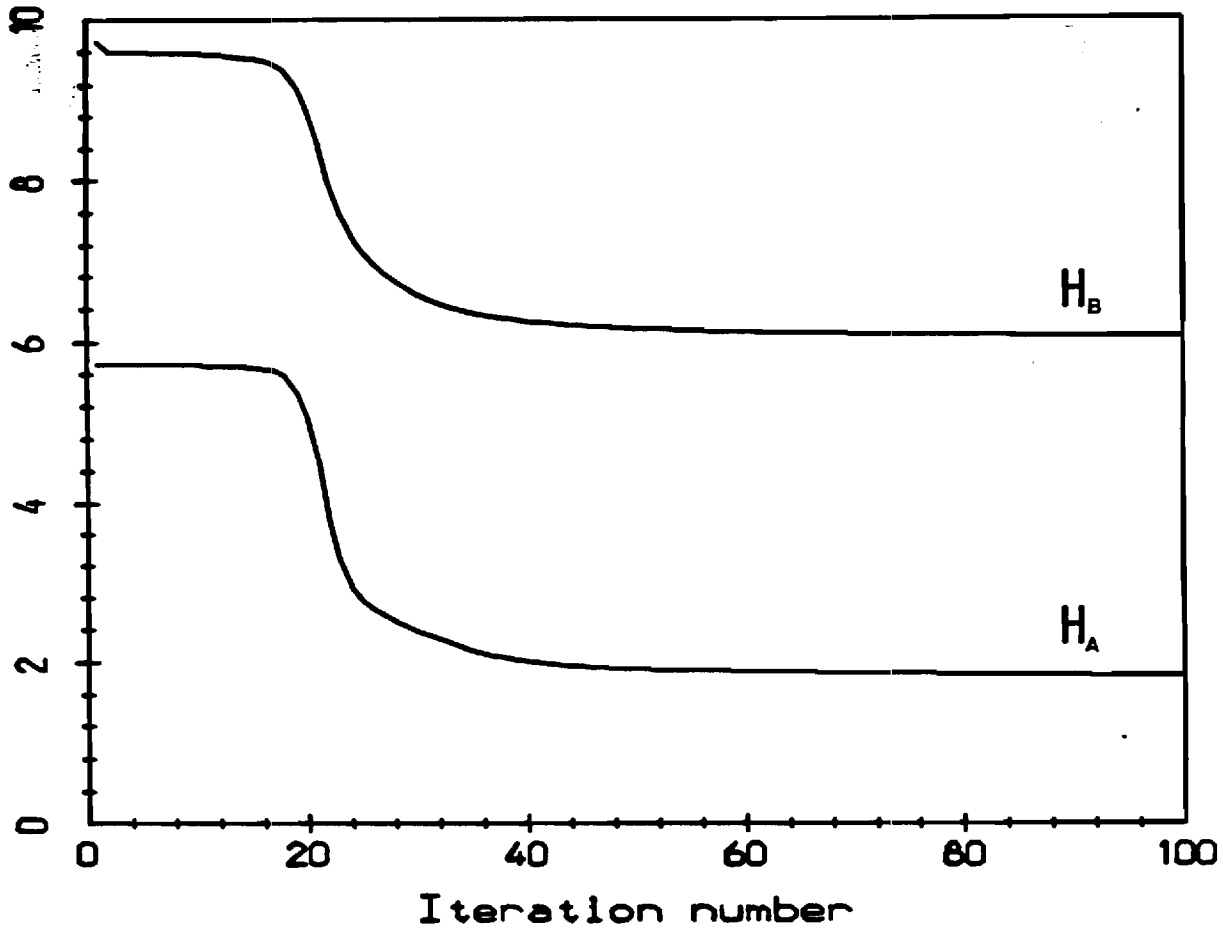


Figure 6.2: Evolution of the entropies.

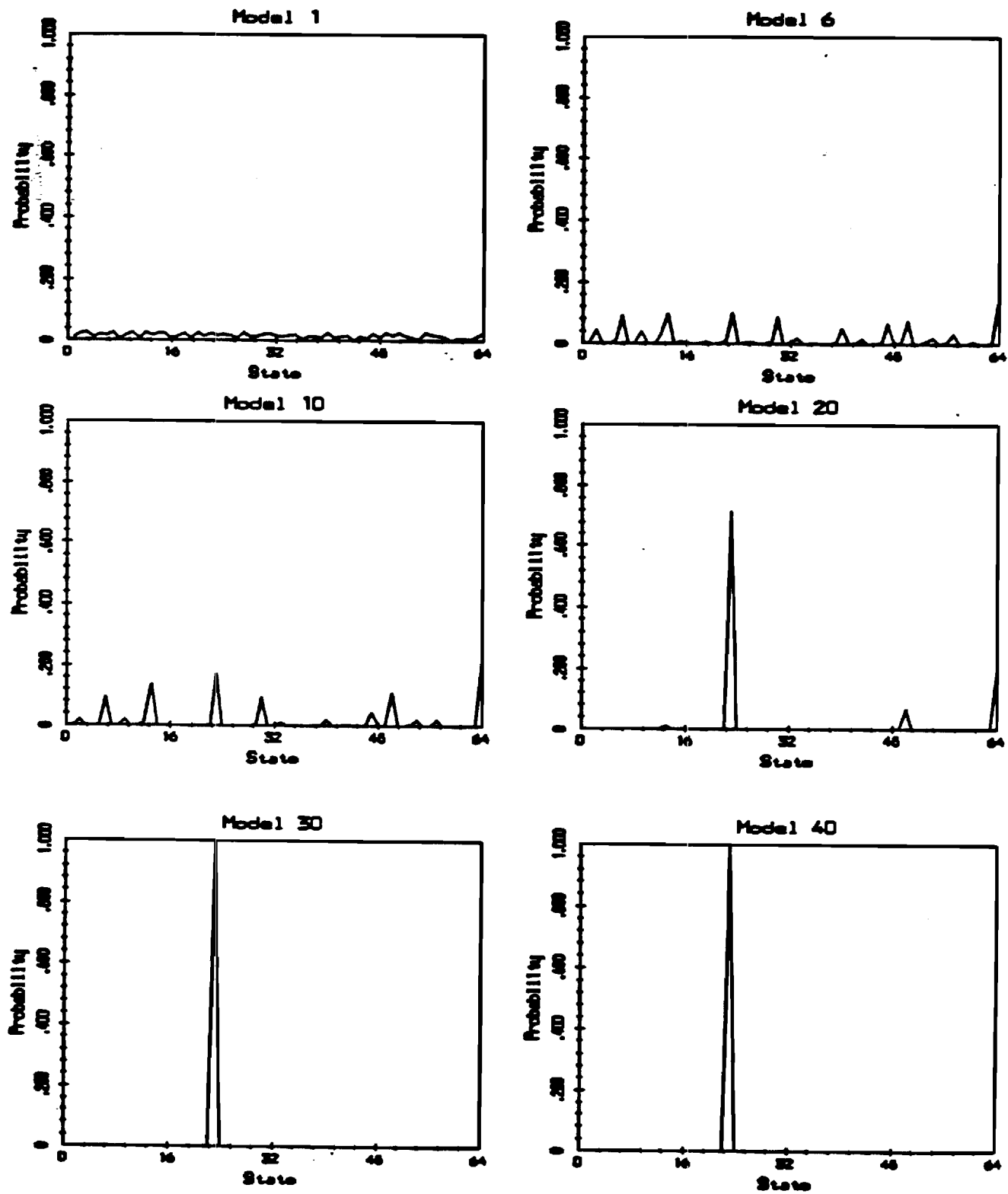


Figure 6.3: Evolution of the initial distribution of the states.

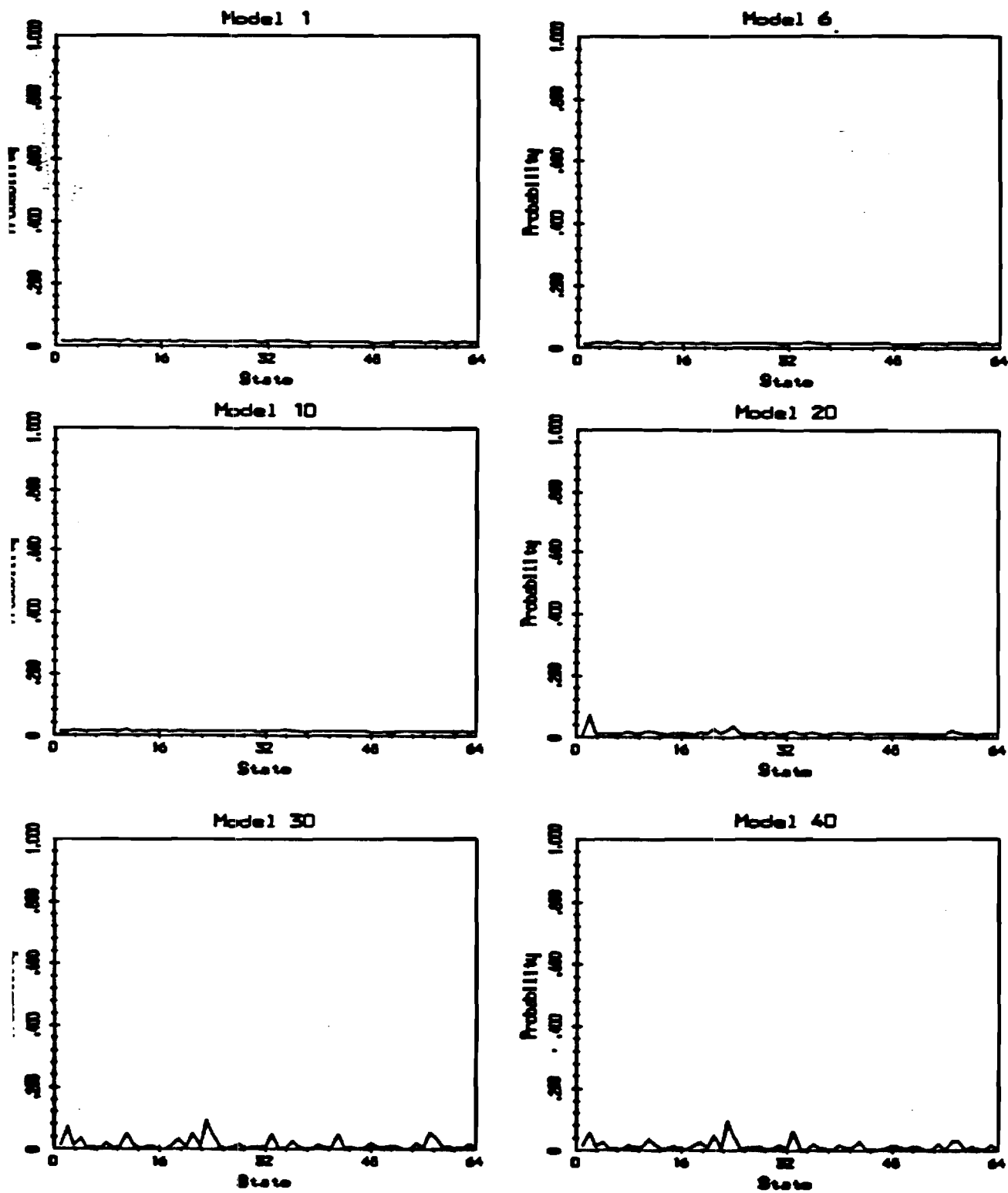


Figure 6.4: Evolution of the steady-state distribution of the states.

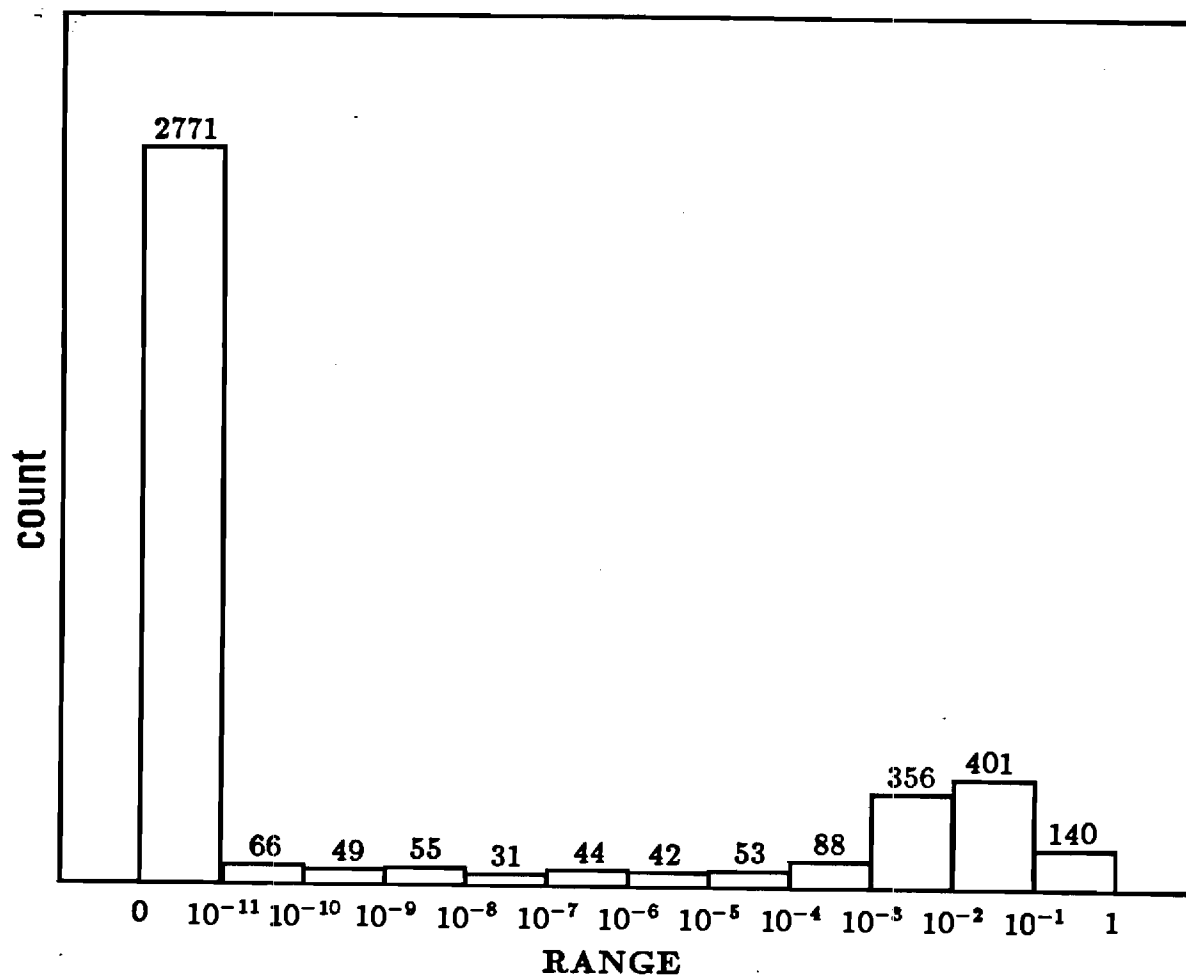


Figure 6.5: Histogram of the entries of the transition matrix A (λ_{100}).

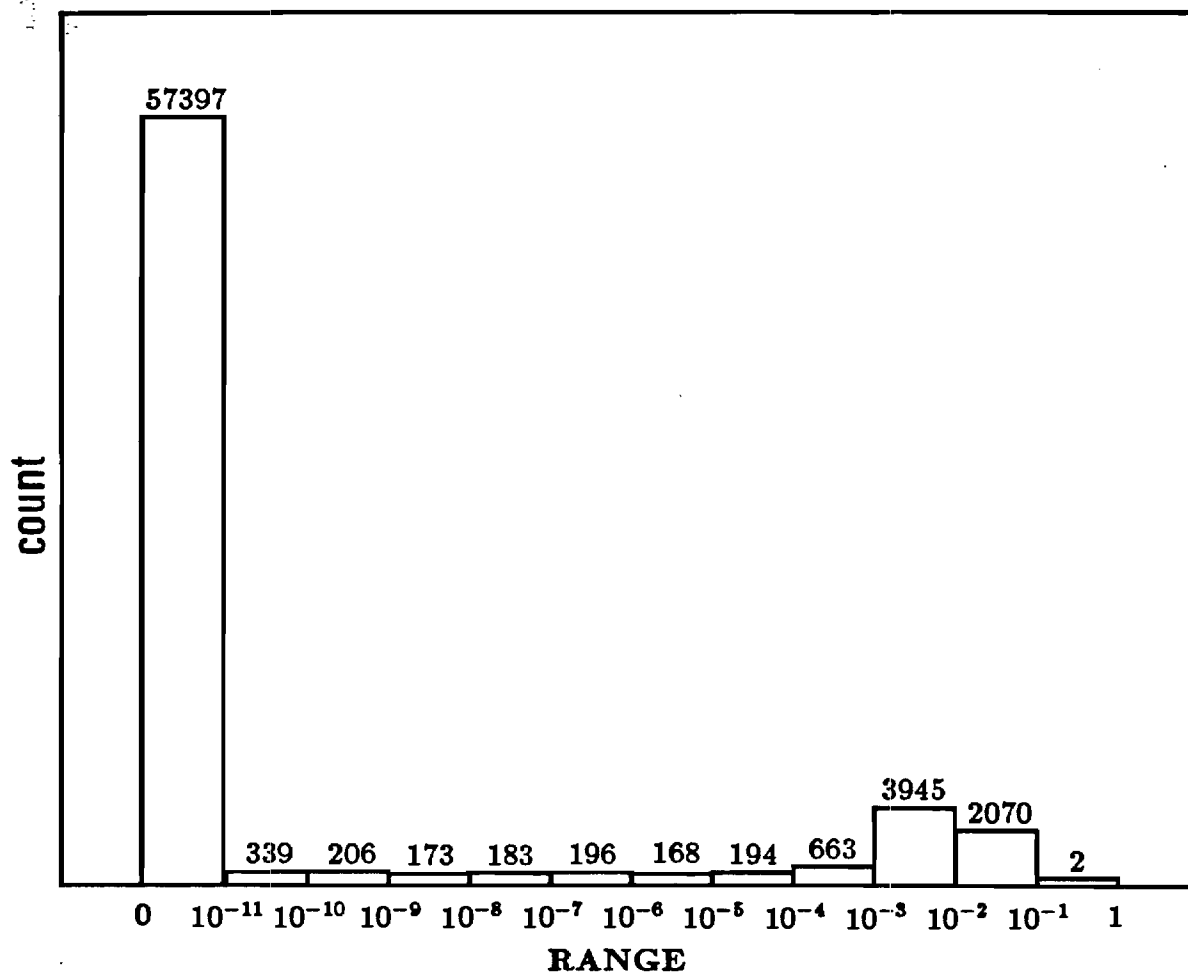


Figure 6.6: Histogram of the entries of the output probability matrix B (λ_{100}).

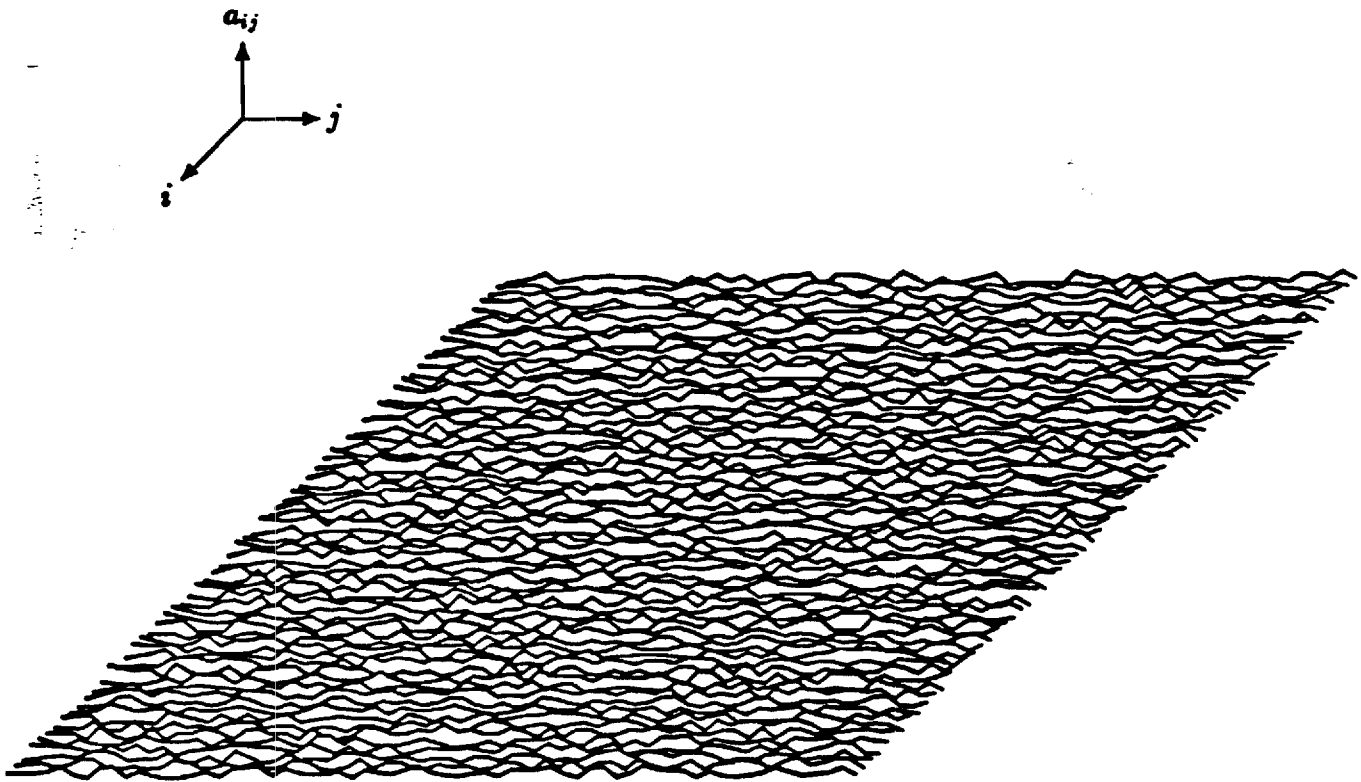


Figure 6.7: Transition matrix of model 1.

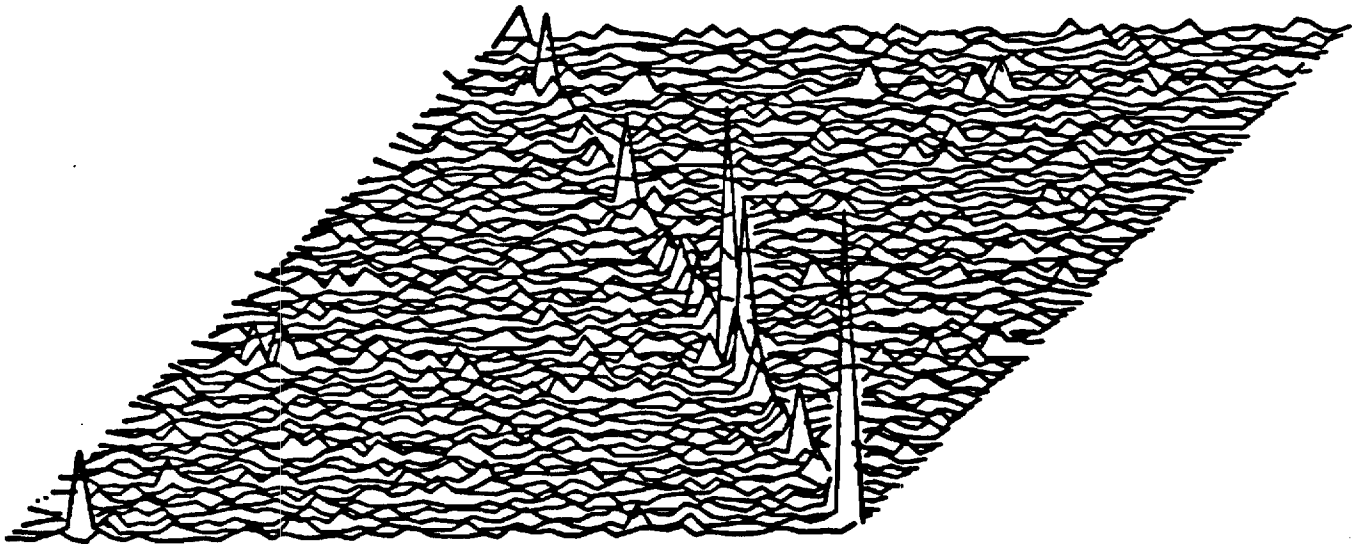


Figure 6.8: Transition matrix of model 20.

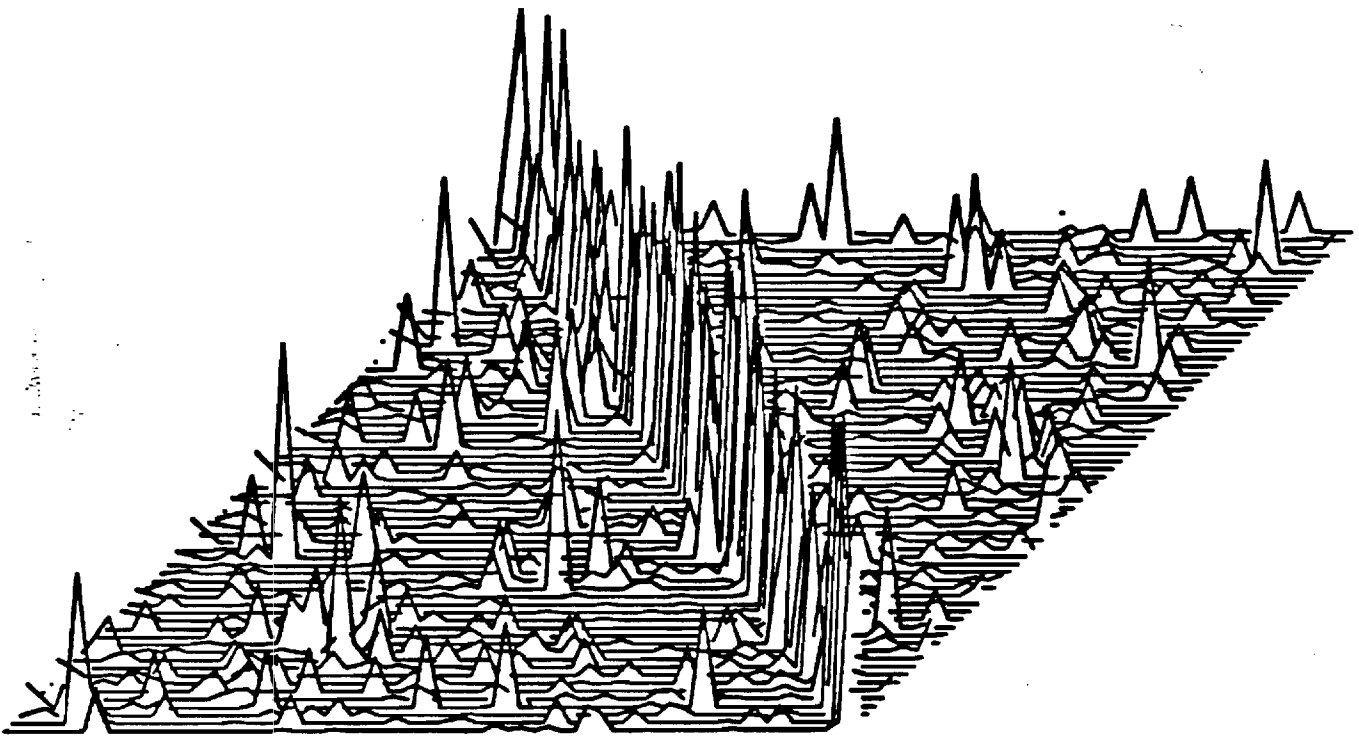


Figure 6.9: Transition matrix of model 40.

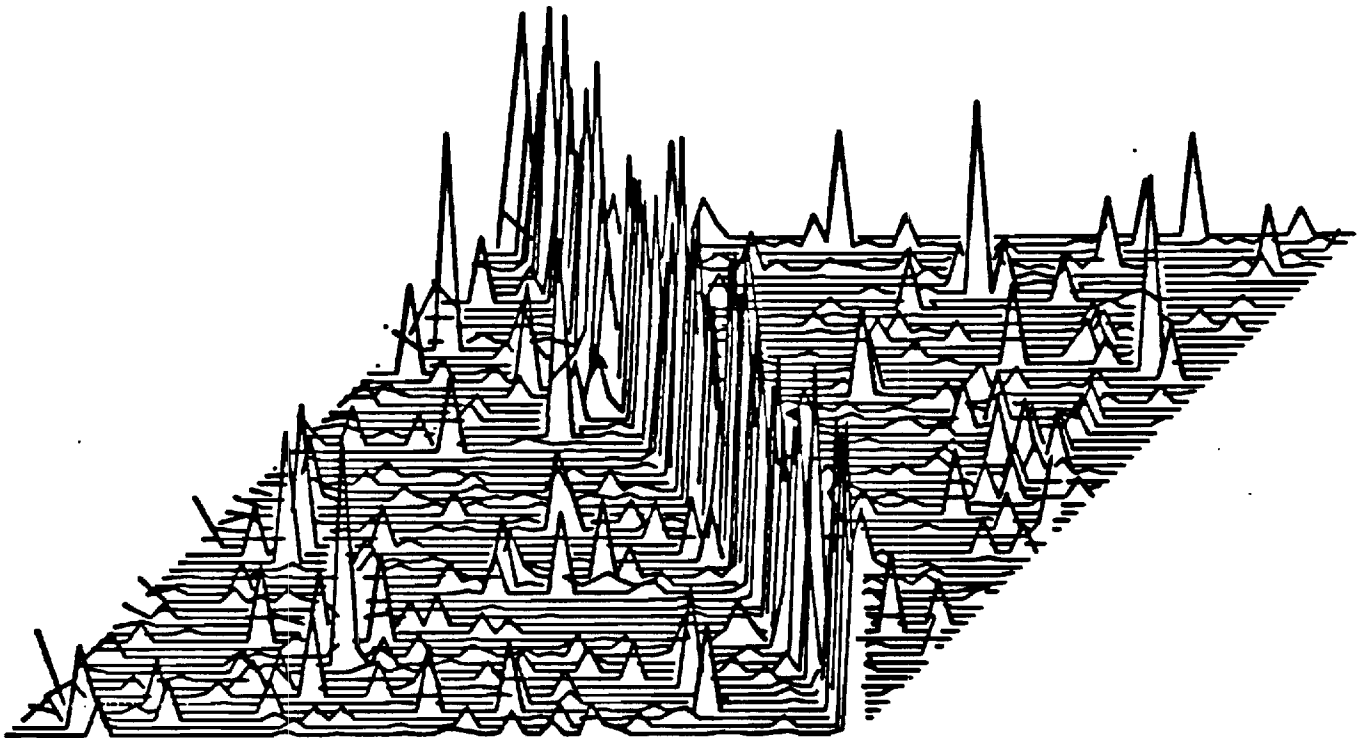


Figure 6.10: Transition matrix of model 100.

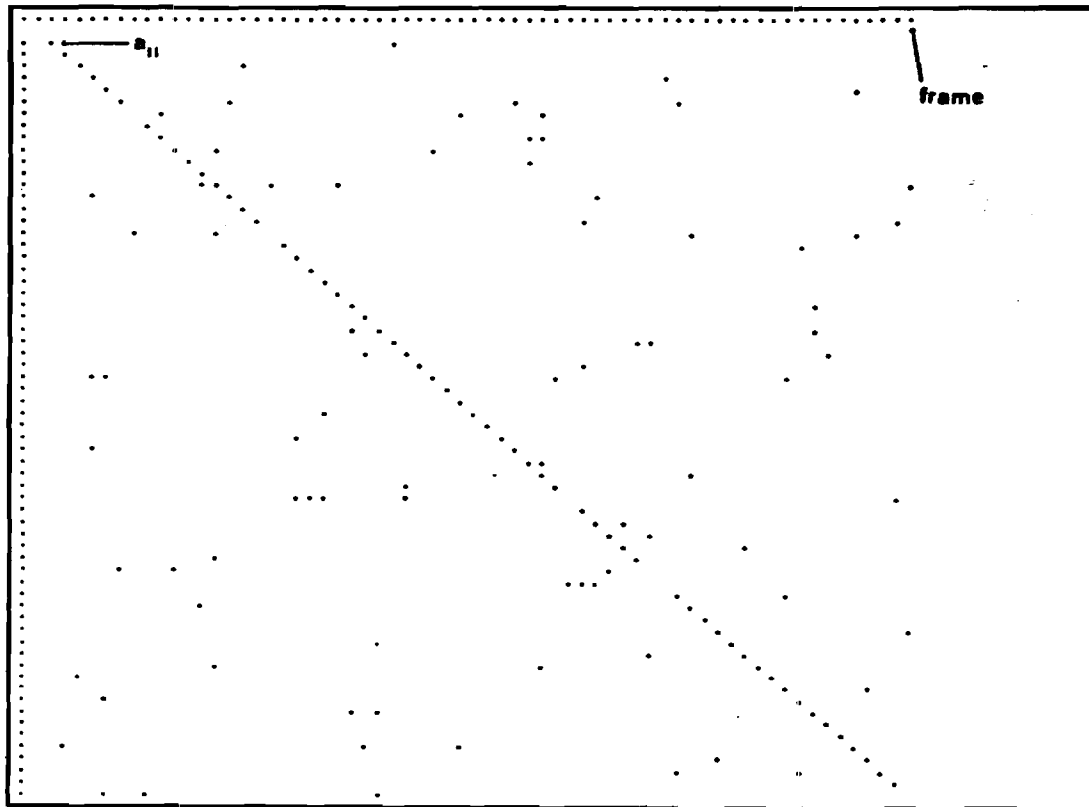


Figure 6.11: The structure of the transition matrix A (λ_{100}).
(dots are entries greater than 0.1.)

6.2 State decoding

The state decoding was carried out on the 15 minutes of speech. Convergence nodes emerged with the distributions shown in figure 6.12. The most probable convergence node weight is 2. The maximum convergence node weight ever encountered (with a very small probability) was 12. In practice one should not expect a weight greater than 8. The most probable convergence lengths are 3 and 4, the maximum convergence length encountered was 28 (very improbable), no conver-

gence length greater than 12 should be expected in practice. The most probable decoder delay is 7 units of time (0.11 second), and generally the decoder delay will not exceed 16 units of time (0.24 second).

The most probable log-magnitude spectrum in each of the 64 states is presented in figure 6.13. The expected state spectra are shown in figure 6.14. They introduced a smoothing of the most probable spectra (flattening the peaks). This will cause the reconstructed "expected speech" to be a little less clear than the "most probable speech," but will smooth out the quirky sound effects produced by the vector quantizer.

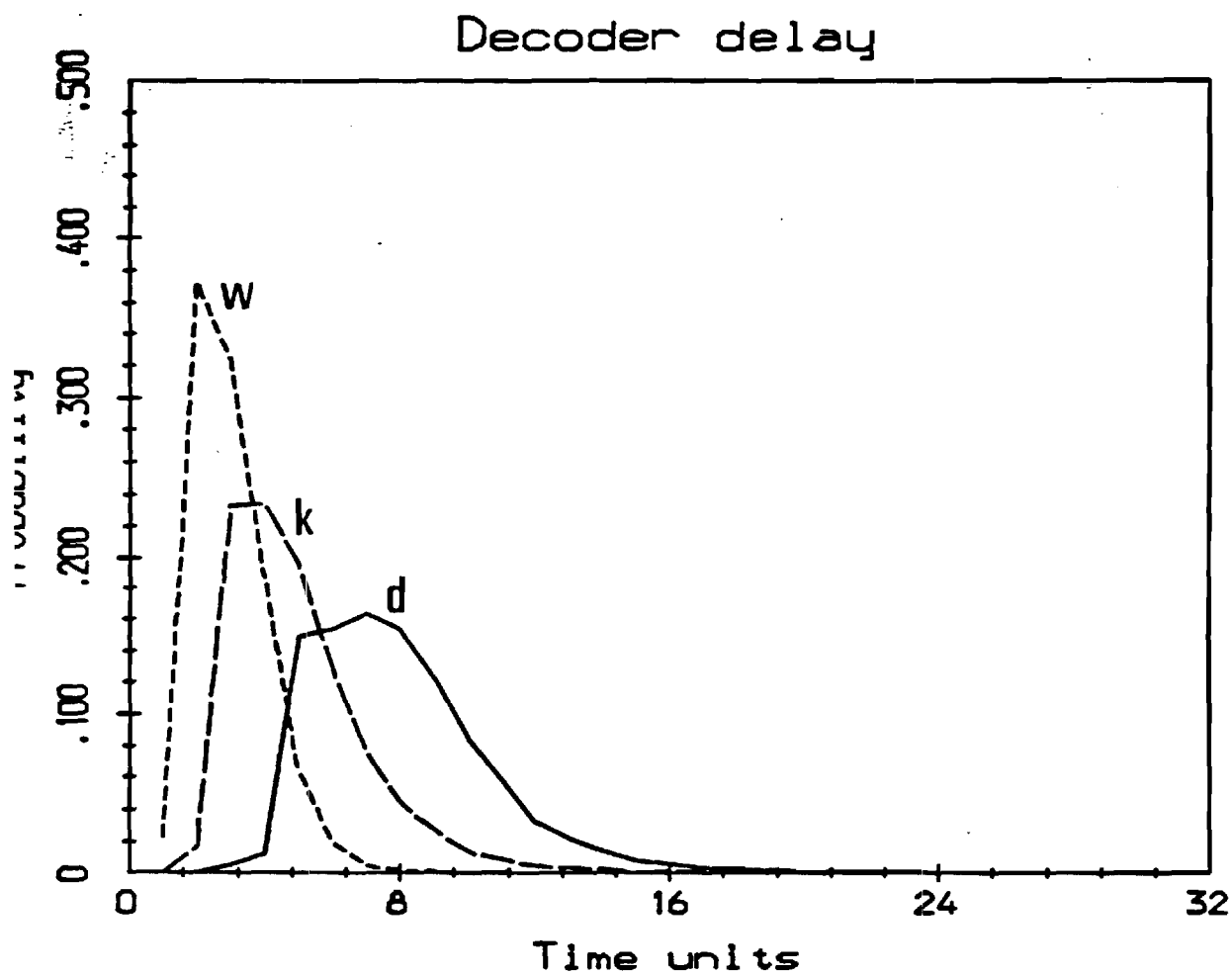


Figure 6.12: Distributions of the convergence node weight, convergence length, and decoder delay.

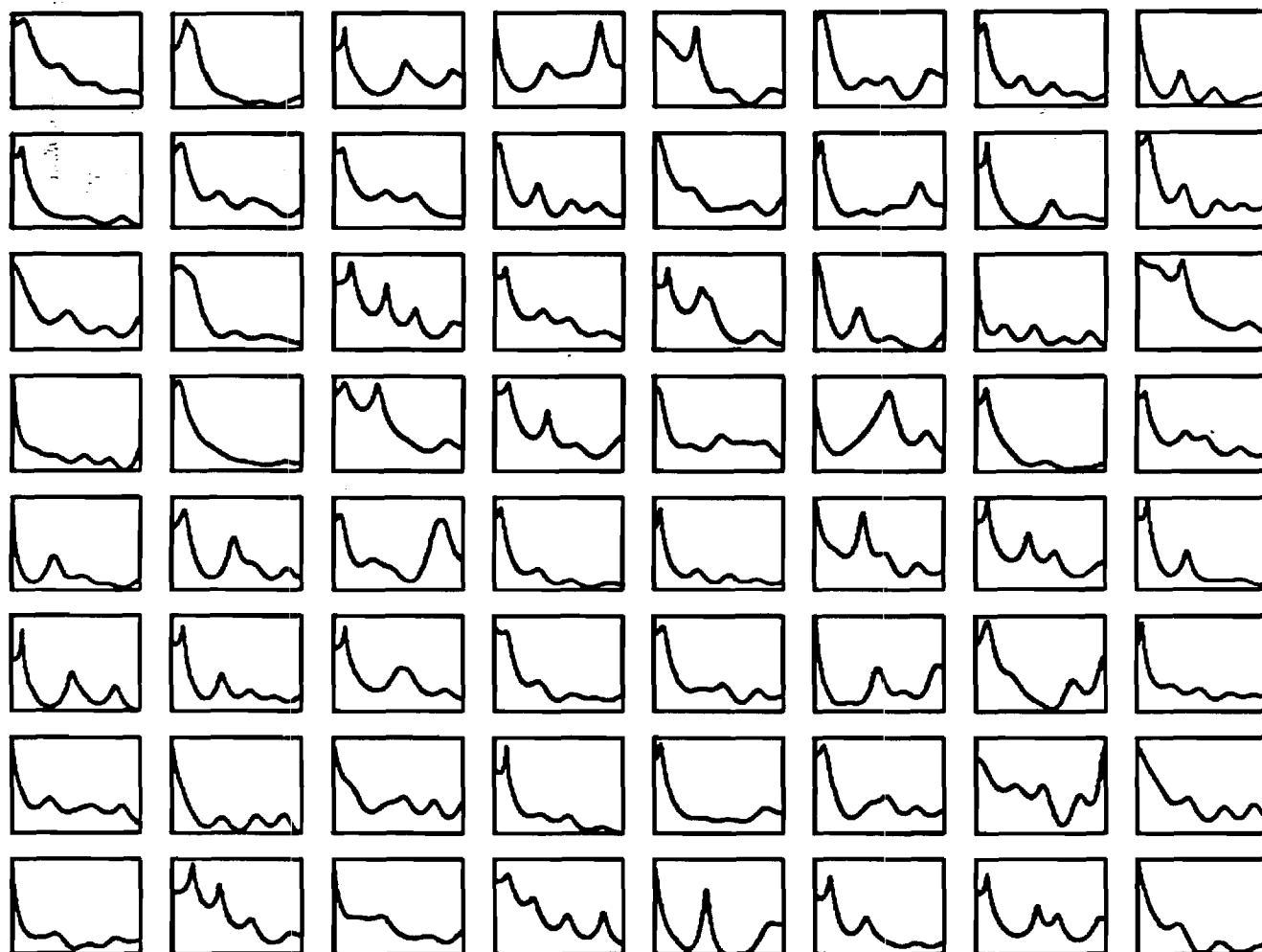


Figure 6.13: Most probable log-magnitude spectrum in each of the 64 states.

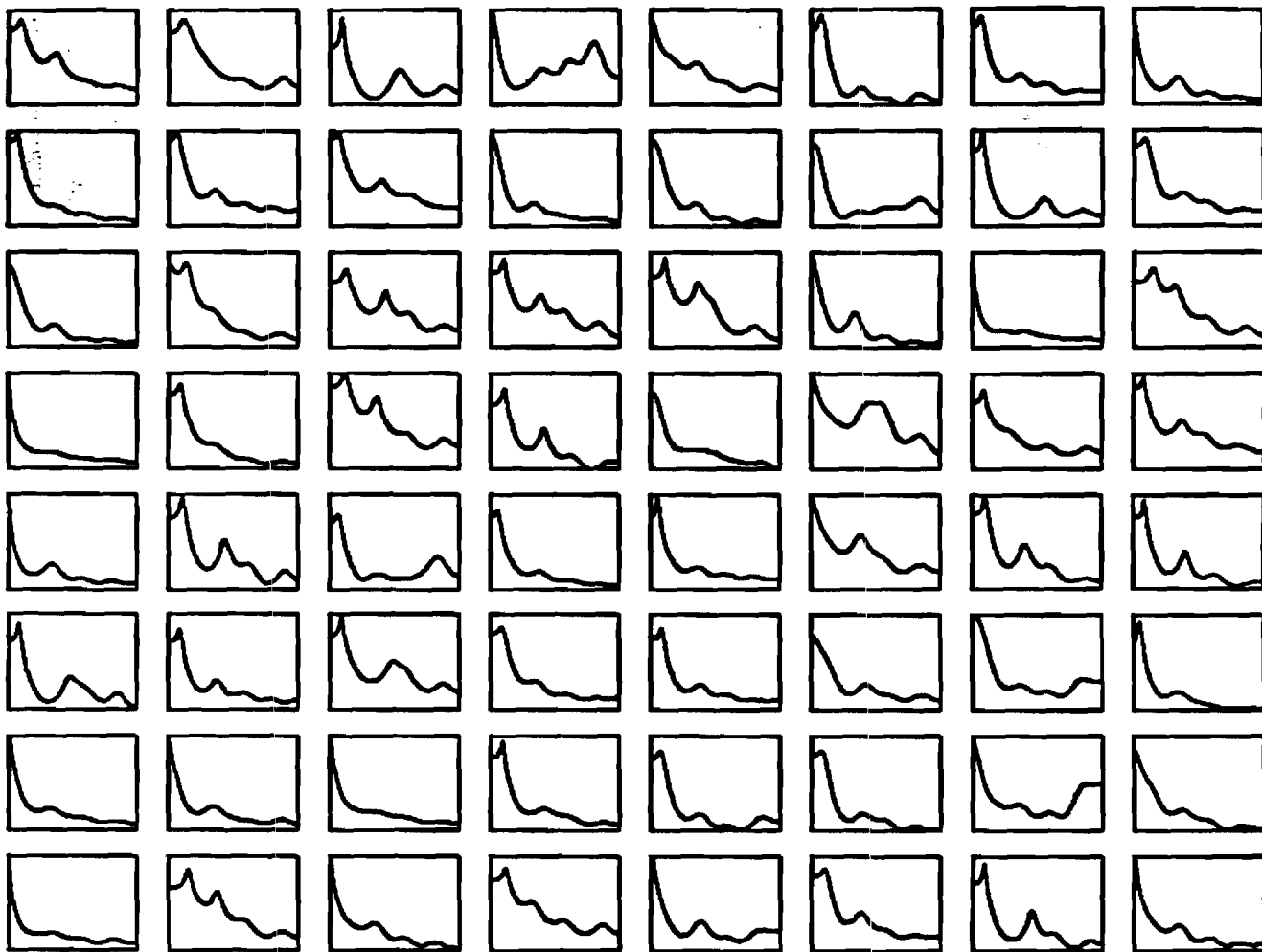


Figure 6.14: Expected log-magnitude spectrum in each of the 64 states.

6.3 Speech coder bit rate

The bit rate of the Markov-VQ coder is given in table 6.3. For every codebook size in the table, a VQ codebook was generated from the 15 minutes of speech. The speech was then vector quantized according to these different codebooks, providing the different speech "training" sequences of LPC vectors used to evaluate the bit rate (entropy H_A) of the coder, as explained in section 5.1. The Markov-VQ shows a significant reduction of the coder bit-rate over the classical VQ (35–47% reduction). As will be seen later, an even larger bit rate reduction (for roughly the same distortion) is possible with the HMM-VQ coder. The speech intelligibility produced by the Markov-VQ is discussed in section 6.4.

The bit rate of the partitioned HMM-VQ is directly derived from the entropies H_A and H_B . As described earlier, these entropies can be computed directly from the model λ (see section 6.1 for the numerical values), or they can be estimated by frequency counts on the sequences X and Y , once the state sequence X has been decoded (given the model λ). Table 6.6 shows the entropies H_A and H_B derived by these two methods. The entropies H_A and H_B directly derived from the sequences X and Y are more appropriate to describe the bit rate of the partitioned HMM-VQ, which is 7.68 bits/frame in that case. The partitioned HMM-VQ coder, even though it encodes the same exact information Y as the 1024-level VQ coder, achieves a lower bit rate than the 1024-level VQ (10 bits/frame), and a higher bit rate than the 1024-level Markov-VQ (5.3 bits/frame). Therefore if we are only concerned with designing a very-low-bit-rate speech coder, with a bit rate in the range $]5\text{b/f}, 10\text{b/f}[$, the Markov-VQ provides a simpler and more efficient design than the partitioned HMM-VQ.

Now if we are concerned with designing a speech coder with a bit rate below 5 bits/frame, we need to use the HMM-VQ coder with a pegging period K sufficiently

long. For example, to be more efficient than the 5.3 bits/frame Markov-VQ, the HMM-VQ needs to use a pegging period $K \geq 3$ (need $K \geq 4$ to do better than a 64-level Markov-VQ) — see tables 6.3 and 6.4, and figure 5.1. The quality of the speech synthesized by the HMM-VQ coder will be compared to the quality of the Markov-VQ speech in section 6.4.

Table 6.3: Markov-VQ bit rate.

Codebook Size	Bit rate (bits/frame)		Bit rate reduction (%)
	VQ	Markov-VQ	
1024	10	5.3	47
64	6	3.9	35
32	5	3.2	36
16	4	2.4	40
8	3	1.7	43
4	2	1.0	50
2	1	0.4	56

Table 6.4: Bit rate as a function of the pegging period K ($H_A = 1.68$, $H_B = 5.99$).

K (frames)	2	3	4	5	6	7	8
B(K) (bits/frame)	7.67	4.68	3.68	3.18	2.88	2.68	2.54
K (frames)	9	10	11	12	13	14	15
B(K) (bits/frame)	2.43	2.35	2.28	2.22	2.18	2.14	2.11

Table 6.5: Bit rate as a function of the pegging period K ($H_A = 1.84$, $H_B = 6.07$).

K (frames)	2	3	4	5	6	7	8
B(K) (bits/frame)	7.9	4.85	3.83	3.33	3.02	2.82	2.67
K (frames)	9	10	11	12	13	14	15
B(K) (bits/frame)	2.56	2.48	2.41	2.35	2.31	2.27	2.24

Table 6.6: Entropies of the HMM-VQ coder from two methods of derivation.

Method of derivation	Entropies	
	H_A	H_B
from model λ_{100}	1.8386	6.0732
from sequences X and Y	1.6783	5.9983

6.4 Speech intelligibility and observation decoding

The 1024-level vector quantizer will be used as a reference for speech intelligibility and quality comparisons between the different types of speech coders investigated. The bit rate needed by the 1024-level VQ to achieve this reference speech quality is 10 bits/frame. It can be reduced to 5.3 bits/frame with a Markov-VQ (see table 6.3). The bit rates achievable by VQ's and Markov-VQ's with smaller codebooks are given in table 6.3. Speech intelligibility evaluation is carried out on the following test sentence:

“Before we undertake the wearisome task of analyzing british humor it would be well to define our terms.”

This sentence used for the testing was recorded by the same male American speaker who recorded the speech training database three years earlier. Even though the actual speech waveform of the test utterance was not in the training database, the “text” of this sentence was also included in the training database. The recording of the training database was performed under a more noisy environment than the one for the test sentence. Part of the test speech waveform is shown in figures 6.15, 6.16, and 6.17.

The pitch excited LPC-VQ introduces a noticeable speech distortion when compared to the non-quantized LPC speech, even for a 1024-level codebook. The smaller the codebook size, the greater the distortion. The most noticeable distortion seems to be localized “quirky” sound effects, probably mostly due to the fact that the VQ does not ensure a frame to frame spectral continuity. The speech distortion introduced by the different codebook size VQ's is evaluated against the non-quantized LPC speech and the 1024-level vector quantized speech by means of the LPC log-likelihood distance in table 6.7. A subjective evaluation of the speech

quality was conducted by seven listeners. These results are shown in table 6.8.

The speech produced by the partitioned HMM-VQ is identical to the speech produced by the 1024-level VQ and the 1024-level Markov-VQ (with the respective bit rates 7.68, 10, and 5.3 bits/frame). Therefore the speech quality evaluation of the preceding paragraph also applies to the partitioned HMM-VQ coder.

The “most probable” HMM-VQ coder (see chapters 4 and 5) requires a bit rate of 1.68 bits/frame. To achieve a comparable low bit rate, a VQ would require a 4-level codebook, and a Markov-VQ an 8-level codebook. With so few vectors in the codebook the VQ speech is expected to be of very poor quality, if intelligible at all. However, it turns out that the speech generated by the “most probable” HMM-VQ is very much intelligible, and in any case of much better quality than the VQ speech at a comparable bit rate. This is certainly a very interesting result. The “expected” HMM-VQ coder, with the same bit rate as the “most probable” HMM-VQ, produces smoother speech spectra (see figures 6.13 and 6.14). As a result the speech is a little less clear, but better in terms of overall speech quality because the “quirky” sound effects, mainly produced by the vector quantization, are smoothed out.

The speech intelligibility and quality of the “simpler” HMM-VQ (like “most probable” and “expected” HMM-VQ coders) can be improved by the “more sophisticated” HMM-VQ (with a reconstruction scheme based on the ML-TDA). The degree of improvement is a function of the pegging period: if the pegging period K is too small (the minimum being $K = 2$), the TDA does not have enough “context” to operate optimally; if the pegging period K is too large the long term capability of the TDA to predict (without any clue about the actual speech) becomes less reliable — and equivalent to the reconstruction capability of the “most probable” HMM-VQ coder. The optimum pegging period should be

expected to reflect the optimum state-decoding length of independent blocks, as described earlier by the convergence length, convergence node weight, and delay of the state-decoder — probably suggesting an optimum pegging period between 3 and 12 frames. Examples of reconstructed spectra and speech waveforms are given in figures 6.18 and 6.19.

The subjective speech quality was evaluated by seven untrained listeners with the Paired Acceptability Rating Method [147,150]. Two sessions were used to evaluate nine systems of coders. The same sentence coded by two different systems is played to the listener who is asked to give a score in the range [0,100] for each system. All the possible pairs (without taking into account a pair ordering) are played. The listener listens to two reference systems played in pair twice prior to the beginning of the test: he is asked to base his scores on a high quality anchor rated 80 (pitch excited LPC) and a low quality anchor rated 20 (4-level pitch excited LPC-VQ). Results are shown in table 6.8. The significance matrix show if systems in the left column are significantly different in terms of speech quality from better systems in the upper row. The “most probable” and “expected” speech (1.68 bits/frame) were rated much better than the 4-level VQ (2 bits/frame). The “expected” speech was preferred over the “most probable” speech because it smoothed out the “quirkiness” due to the VQ. The HMM coders with a reconstruction based on the TDA were better than the “most probable” and “expected” speech. The HMM speech coders with different pegging periods K (i.e., different bit rates) were not distinguishable in terms of speech quality and were also almost indistinguishable from the 1024-level VQ.

Table 6.7: LPC log-likelihood distances for the test sentence.

Coder type	Distance to 1024-level VQ	Distance to unquantized LPC
VQ 1024	0	88.2
VQ 64	59.9	134.5
VQ 32	64.1	144.8
VQ 16	78.6	164.8
VQ 8	104.3	198.9
VQ 4	130.5	232.3
VQ 2	186.4	292.6
HMM $K = 3$	39.1	129.8
HMM $K = 4$	54.6	145.7
HMM $K = 6$	63.7	155.3
HMM $K = 8$	67.9	159.6
HMM exp.	64.3	169.5
HMM most p.	84.3	181.6

Table 6.8: Results of the subjective quality testing.

Systems: Session 1	
System	Speech/Coder type
A1	pitch excited LPC
A2	1024-level VQ
A3	HMM "most probable"
A4	HMM "expected"
A5	HMM $K = 3$
A6	4-level VQ

Systems: Session 2	
System	Speech/Coder type
B1	pitch excited LPC
B2	1024-level VQ
B3	HMM $K = 4$
B4	HMM $K = 6$
B5	HMM $K = 8$
B6	4-level VQ

Statistics: Session 1			
Rank	System	Average Score	Standard Deviation
1	A1	59.9	35
2	A2	47.5	28
3	A5	38.7	25
4	A4	34.8	23
5	A3	33.3	23
6	A6	21.1	15

Statistics: Session 2			
Rank	System	Average Score	Standard Deviation
1	B1	58.4	34
2	B2	45.2	28
3	B3	38.0	24
4	B4	37.8	24
5	B5	36.3	23
6	B6	18.1	12

Significance matrix: Session 1						
Systems	1	2	3	4	5	6
1	-	-	-	-	-	-
2	1	-	-	-	-	-
3	2	0	-	0	0	-
4	2	0	-	-	0	-
5	2	0	-	-	-	-
6	2	2	1	1	1	-

Significance matrix: Session 2						
Systems	1	2	3	4	5	6
1	-	-	-	-	-	-
2	1	-	-	-	-	-
3	2	0	-	-	-	-
4	2	0	0	-	-	-
5	2	0	0	0	-	-
6	2	2	2	2	2	-

Significance Level	Meaning
0	no significant difference
1	significant difference 5% probability of error†
2	more significant difference 1% probability of error†

†Error: to say that two coded sentences are significantly different when they are not.

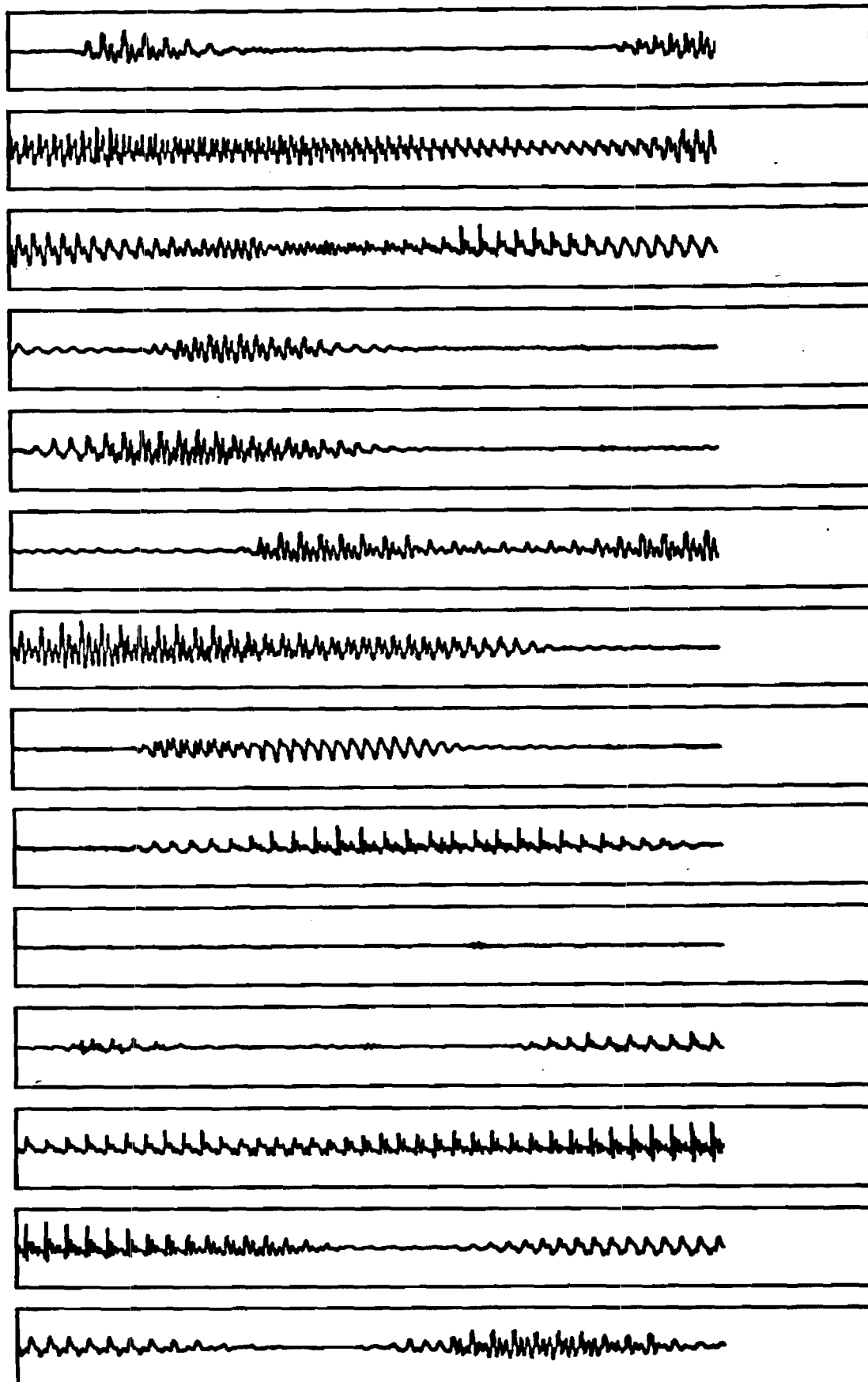


Figure 6.15: Speech waveform of the test utterance (frames 136-373: "Before we undertake the wearisome task of analyzing british...").

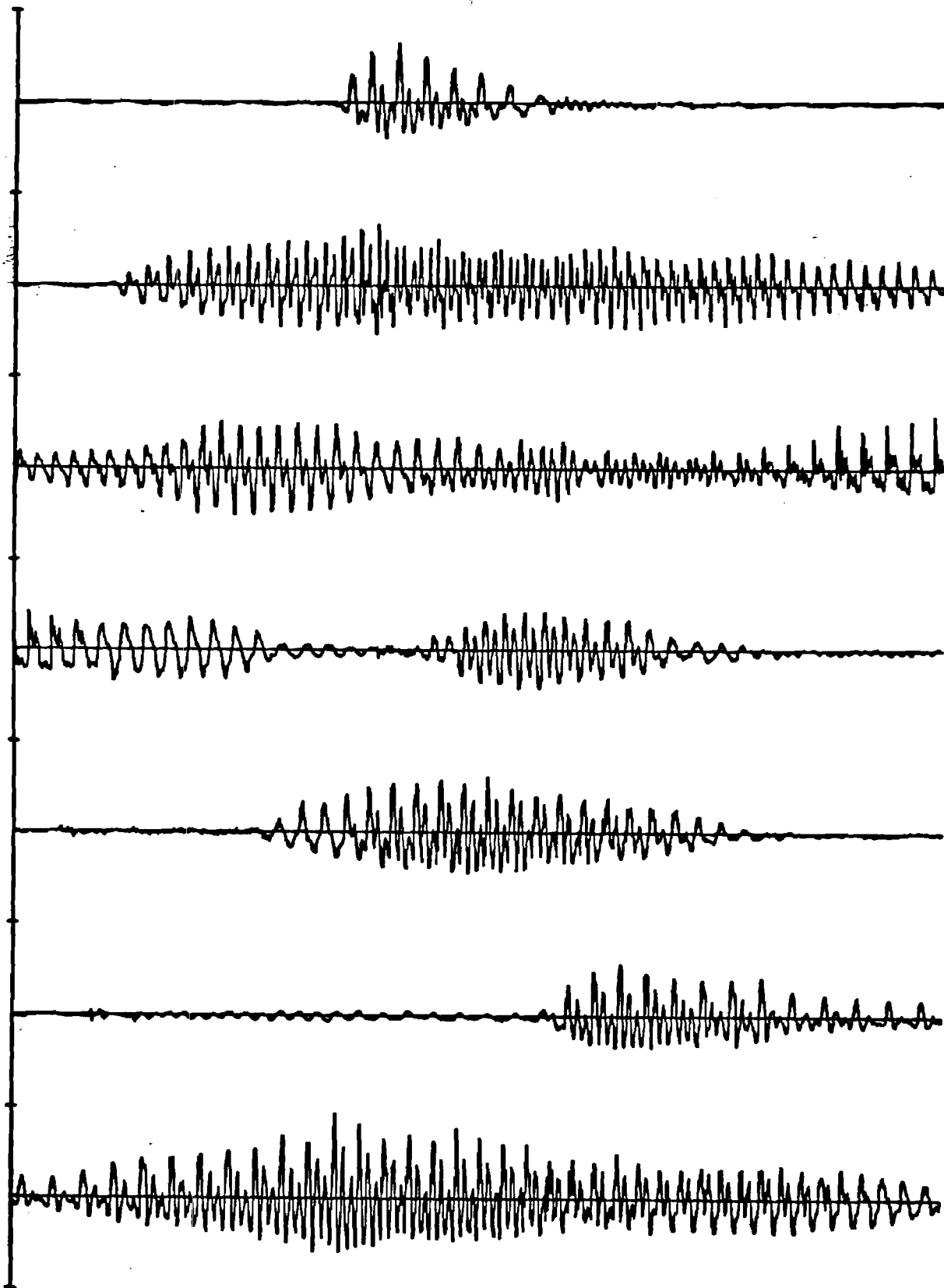


Figure 6.16: More detailed view of the speech waveform (frames 115–233: “Before we undertake the wear...”).

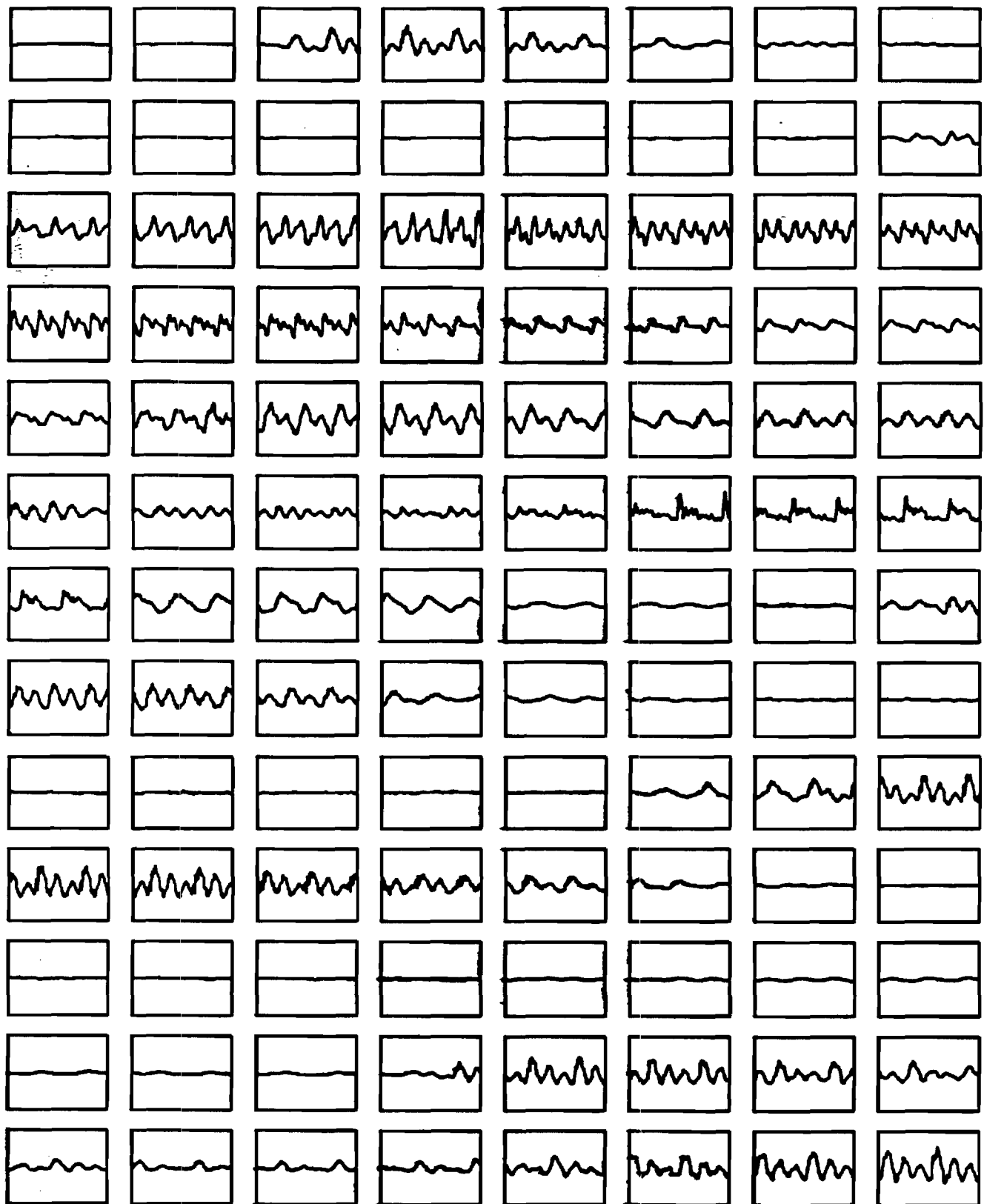


Figure 6.17: Frame by frame view of the speech waveform (frames 120–223: “Before we undertake the wea...”).

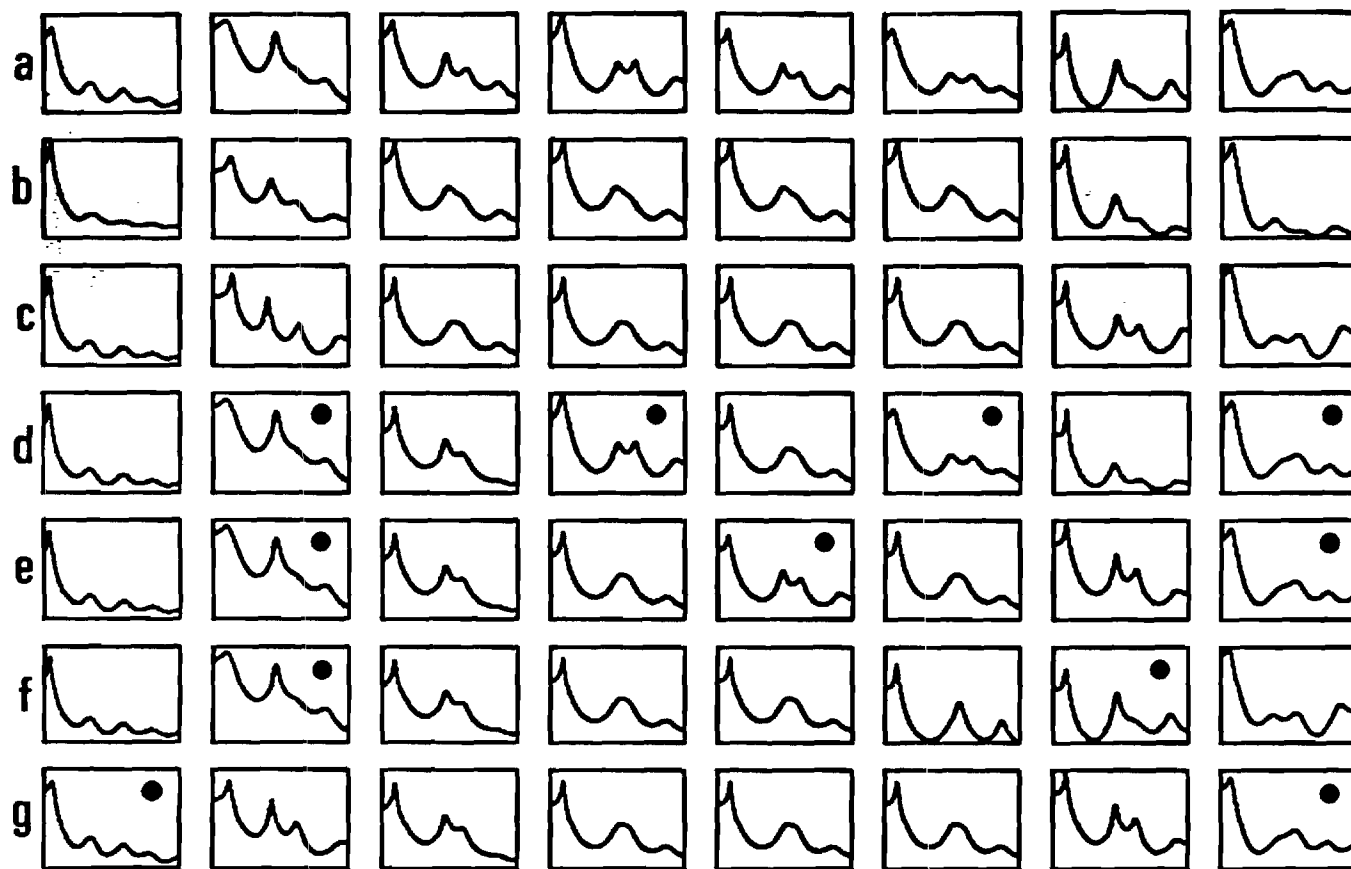


Figure 6.18: Log-spectra of the original and reconstructed speech (frames 120–127).

- a) original 1024-level VQ speech
- b) “expected” reconstructed speech
- c) “most probable” reconstructed speech
- d) HMM ($K = 3$) reconstructed speech
- e) HMM ($K = 4$) reconstructed speech
- f) HMM ($K = 6$) reconstructed speech
- g) HMM ($K = 8$) reconstructed speech

(dots • represent position of pegged observations)

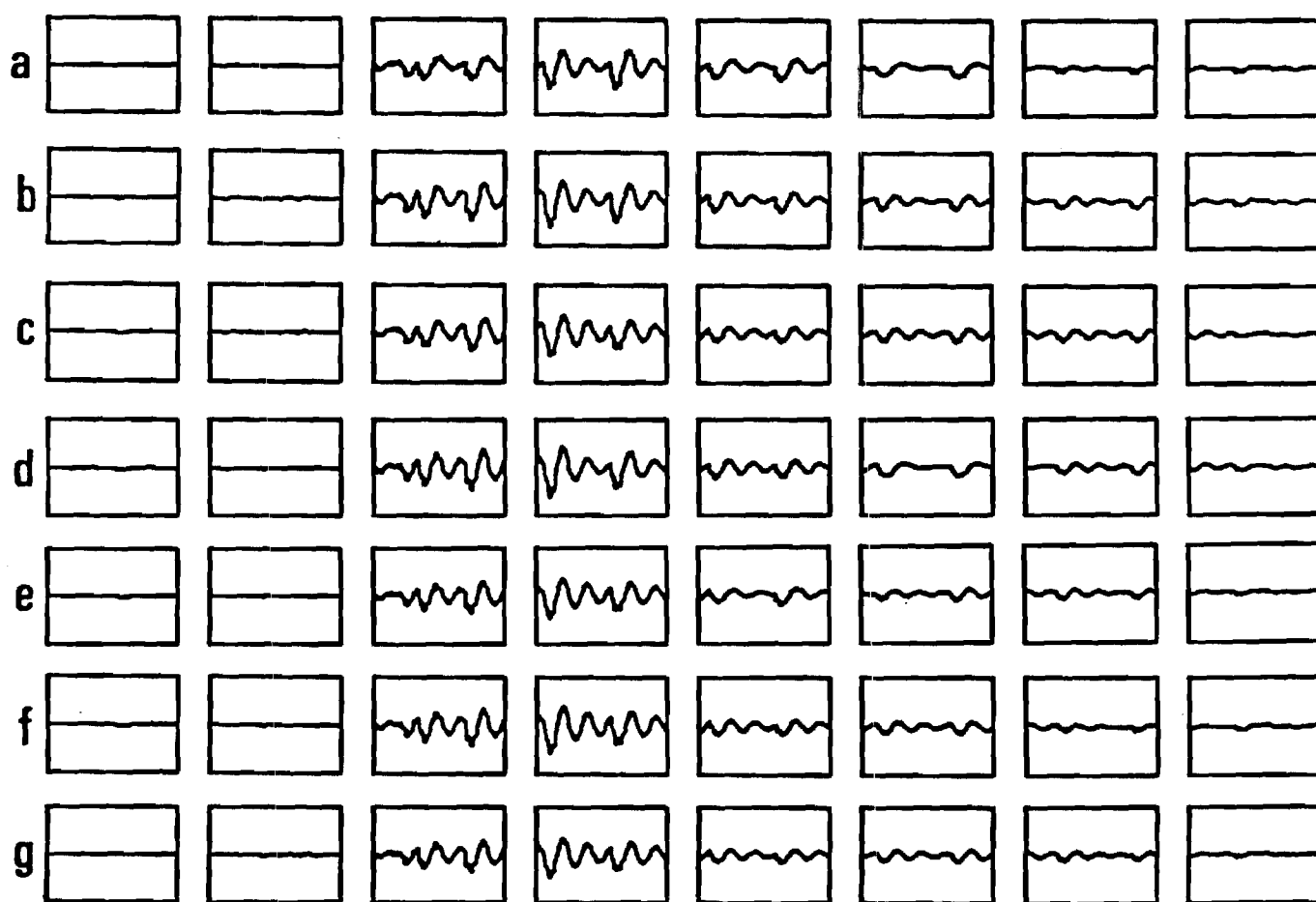


Figure 6.19: Examples of reconstructed waveforms.

(frames 120-127, same labels as in figure 6.18)

6.5 State interpretation

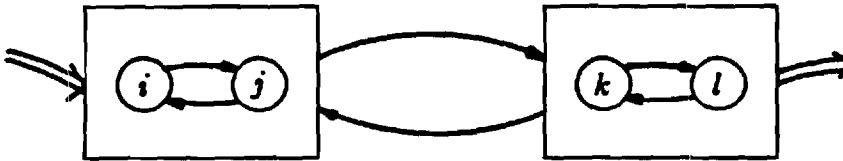
The states of an HMM are a statistical description of the properties and evolutions of the speech signal. They capture the relationships between the various speech events. No further interpretation is needed for the purpose of speech coding. However, for the purposes of speech analysis or speech recognition, a more intuitive interpretation is usually needed. For example, one might want to know how the speech states relate to classical speech sounds like vowels, consonants, phonemes, etc. Some of the typical state structures encountered in speech state sequences are shown in figure 6.20. Some of the speech information is encoded into the nature of the states themselves: states perform an observation clustering (see figure 6.20 a). Typically each state clusters $2^{H_B} \approx 64$ observations. However, most of the speech information is not encoded into the nature of single states, but into the relationships (connections) between states. The simplest type of relationship is encoded into 2-state structures (see figure 6.20 b). This first order relationship can also appear between two blocks of states (see figure 6.20 c) where, in a given block, the states are more or less equivalent — this happens for example with an overspecified system with too many states. The transition between two different 2-state structures will sometimes go through a transient exit state (see figure 6.20 d). The typical structure shown in figure 6.20 e usually models vowels: an onset and an offset state structure surrounds a plateau state structure. The onset and offset depend respectively on the sounds preceding and following the vowel. The plateau corresponds to the steady-state structure of the vowel, and the number of self-transitions $i \rightarrow i$ is directly related to the duration of the vowel. Actually the same state i was found to consistently model the same vowel throughout the speech database. For example, the following plateaux $i = 2$, $i = 52$, and $i = 58$ were found (by segmenting the speech, listening to it, and correlating speech waveforms and state sequences) to systematically represent the vowels /o/ (as in “for”), /i/ (as in “bee”), and /æ/ (as in “hat”) respectively. The formant frequencies F_1

and F_2 were extracted from the VQ observations whenever state 2, 52, or 58 was encountered in the state sequence (for a test waveform of 2000 frames). The $F_1 - F_2$ plot in figure 6.21 shows a clustering of the formants that corresponds to the classical frequency range of the three vowels /o/, /i/, and /æ/. Consonants did not seem to exhibit a plateau structure (the vocal tract does not stay in a steady-state position), therefore their identification in terms of state relationships is harder to perceive. A systematic and thorough interpretation of the state structures and relationships in terms, for example, of phonemes would be needed, and would require a phonetically labelled database and computer time that were not available to us.

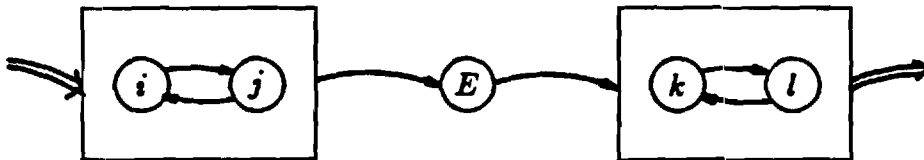
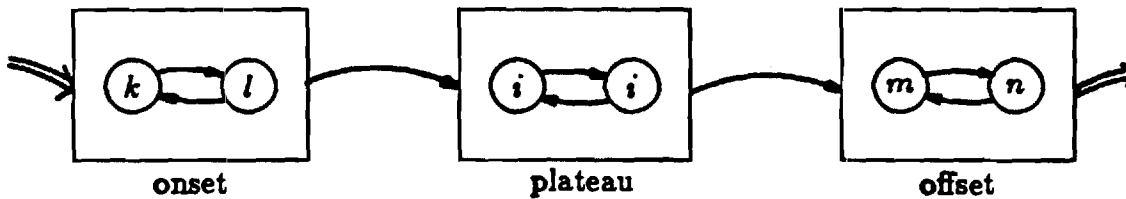
Finally states can also be robustly classified in terms of the following two categories: speech and silence. This can be done by a direct correlation between the state sequence and the speech waveform. It can also be efficiently done by examining the state duration distributions (see figure 6.22). Silence, as compared to speech, is generally made of “long silence/noise” steady-state regions. Silence will be represented by a large number of self-transitions $i \rightarrow i$ encompassing a wide range of state durations. Silence states are therefore identified by “flat” and “stretched” state duration distributions. The following states were identified as silence in different noise environments: 2, 8, 23, 33, and 49. Even though the silence/noise and some unvoiced speech waveforms might be very similar, the corresponding states are different. The HMM states are able to capture the difference between noise and unvoiced speech (like fricatives) because they operate a different clustering for one thing, but mainly because they model two different contexts (speech and silence). As a consequence, a very robust application of this type of HMM is speech segmentation: the state sequence detects the onset and offset of speech very reliably (and therefore the speech and silence segments).



a) State clustering.

b) 2-state structure (possibly $i = j$).

c) Sets of equivalent states.

d) Exit state " E " between two state structures.

e) Vowel-like structure.

Figure 6.20: Some typical state structures.

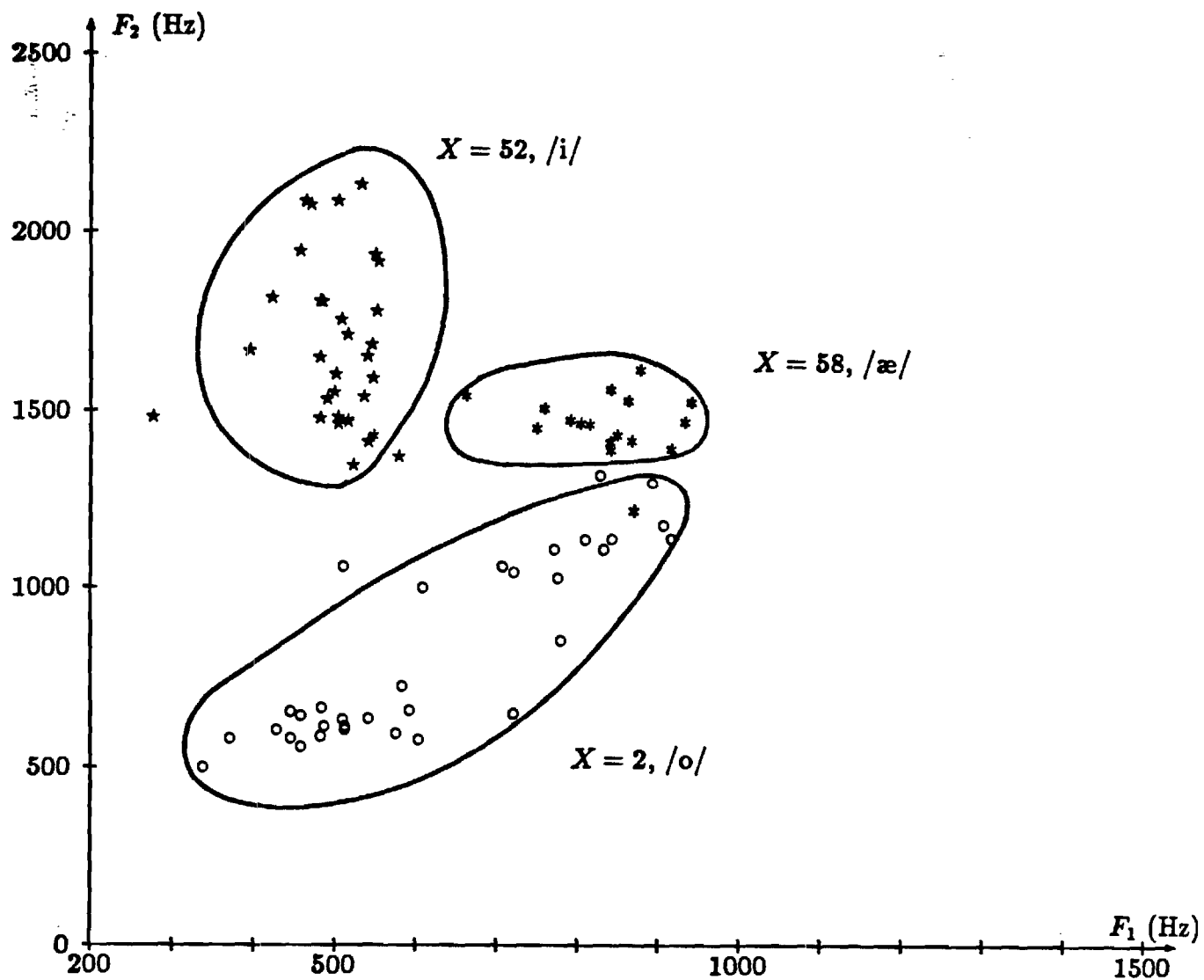


Figure 6.21: Formant clustering of the states 2, 52, and 58.

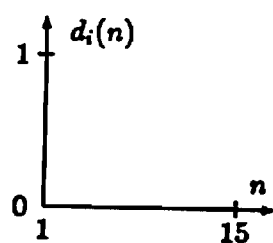
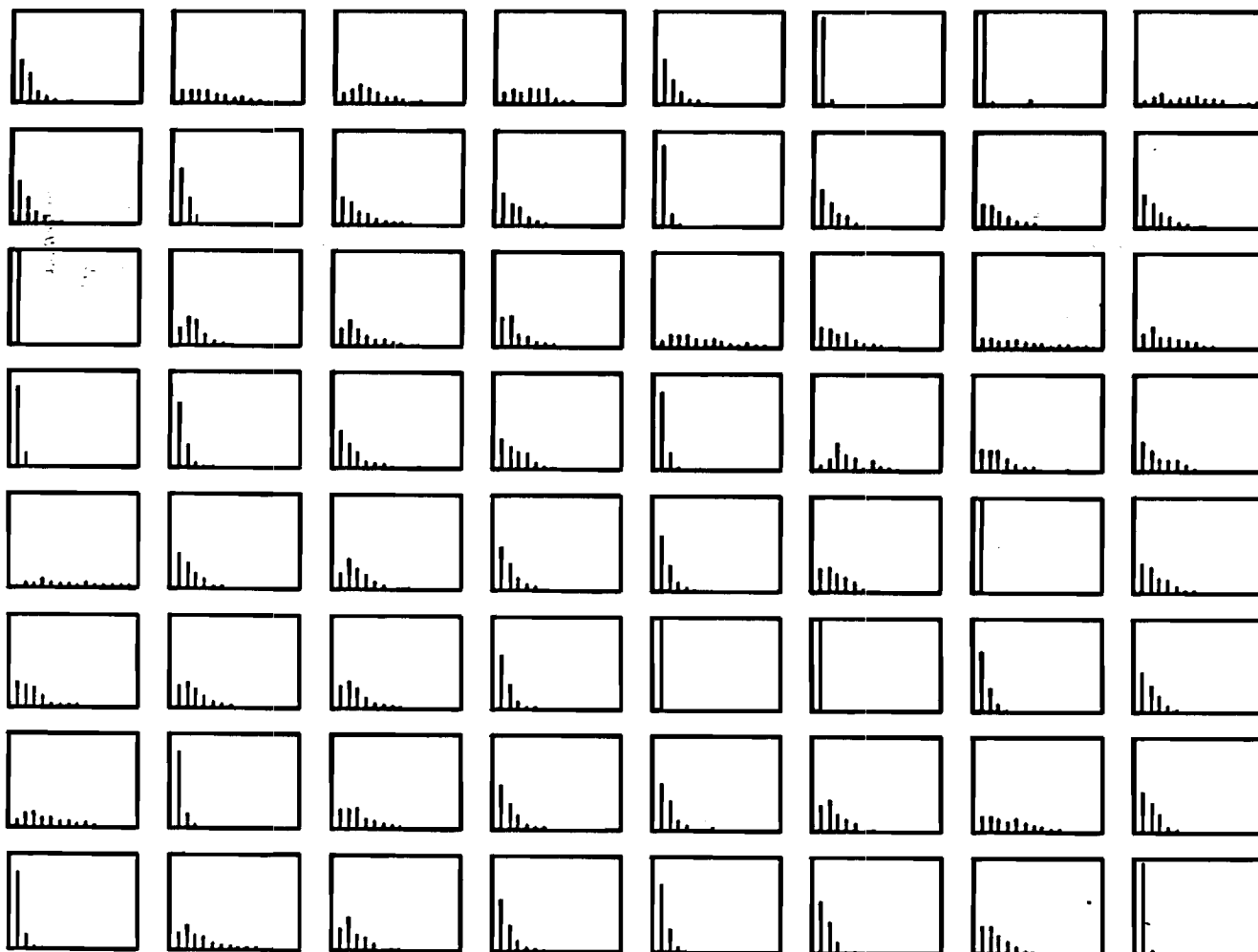


Figure 6.22: State duration distributions.

CHAPTER 7

Conclusion

In this research we stressed the importance of modelling the speech waveform not only as a signal, but also as an observed representation of a hidden language structure. We identified the basic requirements of such an approach, and formulated the general framework of stochastic modelling based on the applicability of hidden Markov models. A consistent mathematical formulation was presented under the unifying approach of maximum likelihood concepts, which were used in the estimation, speech analysis, and speech synthesis phases. Practical algorithms were derived to implement the theoretical model. It was shown that the process of model parameter estimation was practically feasible, even with only a modest computer¹. The model was automatically trained by a computer algorithm operating on a long corpus of 15 minutes of continuous speech. The estimates converged after 100 iterations of the algorithm. The experimental model exhibited a significant structure as modelled by the state and observation entropies. Speech analysis was carried out very efficiently in terms of state sequences. The maximum likelihood trellis decoding with detection of convergence nodes was optimally applied to blocks of speech not exceeding 0.24 second. For speech synthesis, several sub-optimum reconstruction schemes were described and experimented with, operating at bit rates below 8 bits/frame for the encoding of the speech spectral information. Intelligible speech was synthesized from as little information

¹A Data General MV/10000, which is modest compared to a Cray.

as 1.7 bits/frame, using a ML-TDA based on the adjoined HMM. The subjective testing of the synthesized speech quality was provided and compared with (non-hidden) Markov modelling of the speech vectors. An optimum reconstruction procedure was described, but not experimented with: the ML-TDA based on the inverse HMM. Our analysis-synthesis hidden Markov model was shown to be applicable to the problems of speech segmentation, speech analysis, and very-low-bit-rate speech coding/reconstruction. Moreover, a limited investigation of the state interpretation problem showed a good and consistent correlation between typical state sequences and typical speech sounds. For instance, vowels like /i/, /æ/, and /o/ could clearly be identified by their state sequences. The length of the plateau structure in the state sequence directly reflected the duration of the vowel. These preliminary results are promising for the application of our analysis-synthesis HMM to the problems of speech enhancement and continuous speech recognition.

Still, quite a number of topics related to HMM's need to be investigated further through additional research: state interpretation/selection, optimum speech reconstruction, parameter estimation (training), speaker adaptation/independence, acoustical representations, applications, to mention only a few. Regarding the state interpretation problem, a thorough and detailed study is needed. One approach would be, for example, to correlate state sequences and phoneme sequences over the training database. If we were to depart from our approach of a global HMM, and use instead local HMM's of segmented speech units, the state interpretation problem would be partly replaced by the unit selection problem (phonemes, diphones, morphemes, syllables, words?). Concerning the optimum speech reconstruction scheme described in this thesis, practical algorithms would need to be developed, and the validity of the approach should be tested experimentally. In the framework of maximum likelihood parameter estimation, alternate algorithms to the FBA should be developed to improve performance in terms of computing

time and memory requirements. An algorithm which would enable us to update the estimates when new training data becomes available (or to perform adaptive estimation) would be very useful (see [149]). Algorithms not based on a ML approach should also be investigated. Also directly related to the training issues is the problem of speaker adaptation/independence. For example, how can two different HMM's, trained for two different speakers, be "averaged" to generate a speaker-independent HMM? How can a general "speaker-independent" HMM, trained on a large database, be "retrained" (perturbed) on a small database to fit a specific speaker? The HMM is also clearly tied to the initial acoustical representation of the speech waveform. The better the representation, the better the HMM. The more information left out by the acoustical representation, the less reliable the HMM. At best the HMM will capture the structure encoded into the acoustical representation. The acoustical representation is usually given in terms of P-dimensional feature vectors. What feature vectors are appropriate? Energy in frequency bands, LPC-based vectors, mel-cepstrum vectors, etc? In this research the LPC feature vectors were selected from a discrete finite set of possible vectors (the VQ codebook). Further progress in the implementation of VQ's will benefit HMM's. What about an infinite number of possible vectors selected from a continuous support? What continuous probability density functions are the most appropriate? Finally HMM's should be evaluated with respect to their capabilities in a wide range of speech processing applications including speech analysis, coding, synthesis, enhancement, and recognition.

It has been clear throughout this research that hidden Markov models go one step beyond the "previous classical" speech processing techniques. They do so by merging the two viewpoints of signal and language processing into a single conceptual framework. To simplify, HMM's encode the speech information and knowledge into a network of nodes (the states) and connections (the branches) building an HMM trellis. The strengths of the connections between the nodes

are characterized by the model probabilities. The power of this representation also meant an urging need for more computing power. However, the HMM approach has inherent limitations mainly due to the initial assumptions of the model: first-order dependencies and acoustical representation. The Markov chain is a first-order chain. The current observation is directly dependent on the current state, but not on the surrounding observations. As better theoretical descriptions and computing power become available, higher order dependencies will be modelled, and acoustical representations based on the human auditory system will be defined, leading to structures more complex than trellises, such as neural networks [92,131,99].

Bibliography

- [1] S.R. Adke and S.M. Manjunath, "An Introduction to Finite Markov Processes," *John Wiley and Sons*, 1984.
- [2] J. Allen, "A perspective on man-machine communication by speech," *Proceedings of the IEEE*, Vol. 73, No. 11, pp. 1541-1550, November 1985.
- [3] J.B. Anderson, "A stack algorithm for coding with a fidelity criterion," *IEEE Trans. Information Theory*, Vol. IT-20, pp. 211-226, March 1974.
- [4] A. Akmajian, R. A. Demers, and R. M. Harnish, "Linguistics: an Introduction to Language and Communication," *The MIT Press*, Cambridge, Massachusetts, 1984.
- [5] R. Ash, "Information Theory," *John Wiley and Sons*, New York, 1965.
- [6] L.R. Bahl and F. Jelinek, "Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition," *IEEE Trans. Information Theory*, Vol. IT-21, No. 4, pp. 404-411, July 1975.
- [7] L.R. Bahl, J.K. Baker, P.S. Cohen, N.R. Dixon, F. Jelinek, R.L. Mercer, and H.F. Silverman, "Preliminary results on the performance of a system for the automatic recognition of continuous speech," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 425-429, 1976.
- [8] L.R. Bahl, R. Bakis, P.S. Cohen, A.G. Cole, F. Jelinek, B.L. Lewis, and R.L. Mercer, "Recognition results with several experimental acoustic processors," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 249-251, 1979.
- [9] —, "Further results on the recognition of a continuously read natural corpus," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 872-875, 1980.
- [10] L.R. Bahl, F. Jelinek, and R.L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, pp. 179-190, March 1983.

- [11] L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 1, pp. 49-52, 1986.
- [12] J.K. Baker, "The Dragon system — an overview," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-23, No. 1, pp. 24-29, February 1975.
- [13] T.P. Barnwell III, M.A. Clements, S.R. Quackenbush, and E.P. Farges, "Improved objective measurements for speech quality testing," *Final report to the DCA*, DSPL-85-1, 202 pages, January 1985.
- [14] A. Barr and E.A. Feigenbaum, "The Handbook of Artificial Intelligence," *Addison-Wesley Publishing Company, Inc.*, Vol. 1, 1981.
- [15] L.E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Stat.*, Vol. 37, pp. 1554-1563, 1966.
- [16] L.E. Baum and J.A. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology," *Bull. Amer. Math. Soc.*, Vol. 73, pp. 360-363, May 1967.
- [17] L.E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, Vol. 41, pp. 164-171, 1970.
- [18] L.E. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, Vol. 3, pp. 1-8, 1972.
- [19] R.E. Bellman, "Dynamic Programming," *Princeton University Press*, 1957.
- [20] V.K. Bhargava, D. Haccoun, R. Matyas, and P. Nuspl, "Digital Communications by Satellite," *Wiley-Interscience Publication*, New York, 1981.
- [21] A.T. Bharucha-Reid, "Note on estimation of the number of states in a discrete Markov chain," *Experientia*, Vol. 12, pp. 176-177, 1956.
- [22] —, "Elements of Theory of Markov Processes and their Application," *McGraw-Hill Book Company*, New York, 1960.
- [23] B.R. Bhat, "Some properties of regular Markov chains," *Ann. Math. Stat.*, Vol. 32, pp. 59-71, March 1961.
- [24] U.N. Bhat, "Elements of Applied Stochastic Processes," *John Wiley and Sons*, 1984.

- [25] R. Billi, "Vector quantization and Markov source models applied to speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Paris, Vol. 1, pp. 574-577, 1982.
- [26] P. Billingsley, "Statistical methods in Markov chains," *Ann. Math. Stat.*, Vol. 32, pp. 12-40, March 1961.
- [27] D. Blackwell and L. Koopmans, "On the identifiability problem for functions of finite Markov chains," *Ann. Math. Stat.*, Vol. 28, pp. 1011-1015, 1957.
- [28] H. Bourlard, Y. Kamp, and C.J. Wellekens, "Speaker dependent connected speech recognition via phonemic Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 3, pp. 1213-1216, 1985.
- [29] H. Bourlard and C.J. Wellekens, "Connected speech recognition by phonemic semi-Markov chains for state occupancy modelling," *Proc. EUSIPCO*, The Hague, pp. 511-514, 1986.
- [30] A. Buzo, A.H. Gray, Jr., R.M. Gray, and J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, and Signal Processing*, Vol. ASSP-28, No. 5, pp. 562-574, October 1980.
- [31] M.A. Clements and S. Lim, "Hidden Markov model speech recognition based on Kalman filtering," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 1147-1150, 1987.
- [32] E. Charniak and D. Mc. Dermott, "Introduction to Artificial Intelligence," *Addison-Wesley Publishing Company*, 1985.
- [33] X.X. Chen, C.N. Cai, P. Guo, and Y. Sun, "A hidden Markov model applied to chinese four-tone recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 797-800, 1987.
- [34] Y.S. Cheung and S.T. Leung, "Speaker-independent isolated word recognition using word-based vector quantization and hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 1135-1138, 1987.
- [35] P. Chevillat and D.J. Costello, Jr., "An analysis of sequential decoding for specific time-invariant convolutional codes," *IEEE Trans. on Info. Theory*, Vol. IT-24, No. 4, pp. 443-451, July 1978.
- [36] M. Codogno and L. Fissore, "Duration modelling in finite state automata for speech recognition and fast speaker adaptation," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 1269-1272, 1987.

- [37] M. Cravero, L. Fissore, R. Pieraccini, and C. Scagliola, "Syntax driven recognition of connected words by Markov models," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 35.5.1–35.5.4, 1984.
- [38] M. Cravero, R. Pieraccini, and F. Raineri, "Definition and evaluation of phonetic units for speech recognition by hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 3, pp. 2235–2238, 1986.
- [39] —, "Definition of recognition units through two levels of phonemic description," *CSELT Technical Reports*, Vol. XIV, No. 5, October 1986.
- [40] J.R. Crosmer and T.P. Barnwell, III, "A low bit rate segment vocoder based on line spectrum pairs," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 240–243, 1985.
- [41] F.J. Damerau, "Markov Models and Linguistic Theory," *Mouton & Co., Printers, The Hague*, 1971.
- [42] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Ann. of the Royal Statistical Society*, pp. 1–38, 1977.
- [43] C. Derman, "Some asymptotic distribution theory for Markov chains with a denumerable number of states," *Biometrika*, Vol. 43, pp. 285–294, 1956.
- [44] A. Derouault, B. Merialdo, and S. Soudoplatoff, "Phoneme classification using Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 4, pp. 2759–2762, 1986.
- [45] A. Derouault and B. Merialdo, "Natural language modeling for phoneme-to-text transcription," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6, pp. 742–749, November 1986.
- [46] A. Derouault, "Context-dependent phonetic Markov models for large vocabulary speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 1, pp. 360–363, 1987.
- [47] P. D'Orta, M. Ferreti, and S. Scarci, "Phoneme classification for real time speech recognition of Italian," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 1, pp. 81–84, 1987.
- [48] Y. Ephraim, A. Dembo, and L.R. Rabiner, "A minimum discrimination information approach for hidden Markov modeling," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 1, pp. 25–28, 1987.
- [49] R.M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. on Info. Theory*, Vol. IT-9, pp. 64–74, April 1963.

- [50] E.P. Farges and M.A. Clements, "Hidden Markov models applied to very low bit rate speech coding," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 1, pp. 433-436, 1986.
- [51] —, "An analysis-synthesis hidden Markov model of speech," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, New York, 1988, to appear.
- [52] —, "Practical estimation of the parameters of large hidden Markov models," *IEEE Trans. on Information Theory*, 1988, to appear.
- [53] —, "An efficient trellis decoding algorithm," *IEEE Trans. on Communications*, 1988, to appear.
- [54] —, "An analysis-synthesis hidden Markov model of speech: a unified approach," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 1988, to appear.
- [55] —, "Speech analysis by hidden Markov models," *Journal of the Acoustical Society of America*, 1988, to appear.
- [56] G.D. Forney, Jr., "The Viterbi algorithm," *Proc. IEEE*, Vol. 61, No. 3, pp. 268-278, March 1973.
- [57] R.G. Gallager, "Information Theory and Reliable Communication," *John Wiley and Sons, Inc.*, 1968, in particular pp. 63-69 .
- [58] J. Gari, "Some theorems and sufficiency conditions for the maximum likelihood estimator of an unknown parameter in a simple Markov chain," *Biometrika*, Vol. 42, pp. 342-359, 1955.
- [59] E.J. Gilbert, "On the identifiability problem for functions of finite Markov chains," *Ann. Math. Stat.*, Vol. 30, pp. 688-697, 1959.
- [60] L.A. Goodman, "Simplified run tests and likelihood ratio tests for Markoff chains," *Biometrika*, Vol. 45, pp. 181-197, 1958.
- [61] A.H. Gray, Jr. and J.D. Markel, "Distance measures for speech processing," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-24, No. 5, pp. 380-391, October 1976.
- [62] R.M. Gray, A. Buzo, A.H. Gray, Jr., and Y. Matsuyama, "Distortion measures for speech processing," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 367-376, August 1980.
- [63] D. Haccoun and M.J. Fergusson, "Generalized stack algorithms for decoding convolutional codes," *IEEE Trans. on Information Theory*, Vol. IT-21, No. 6, pp. 638-651, November 1975.

- [64] P. Hart, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Systems Science and Cybernetics*, Vol. SSC-4, No. 2, July 1968.
- [65] J.P. Haton, N. Carbonell, D. Fohn, J.F. Mari, and A. Kriouille, "Interaction between stochastic modeling and knowledge-based techniques in acoustic-phonetic decoding of speech," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 868-871, 1987.
- [66] P.G. Hoel, S.C. Port, and C.J. Stone, "Introduction to Stochastic Processes," *Houghton Mifflin Company*, Boston, 1972.
- [67] R.A. Howard, "Dynamic Programming and Markov Processes," *MIT Press*, 1960.
- [68] D.A. Huffman, "A method for the construction of minimum redundancy codes," *Proc. IRE*, Vol. 40, pp. 1098-1101, 1952.
- [69] A. Jarre and R. Pieraccini, "Some experiments on HMM speaker adaptation," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 1273-1276, 1987.
- [70] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM J. Res. Develop.*, pp. 675-685, November 1969.
- [71] F. Jelinek, L.R. Bahl, and R.L. Mercer, "Design of a linguistic statistical decoder for the recognition of continuous speech," *IEEE Trans. Information Theory*, Vol. IT-21, No. 3, pp. 250-256, May 1975.
- [72] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, Vol. 64, No. 4, pp. 532-558, April 1976.
- [73] —, "The development of an experimental discrete dictation recognizer," *Proceedings of the IEEE*, Vol. 73, No. 11, pp. 1616-1624, November 1985.
- [74] B.H. Juang, D.Y. Wong, and A.H. Gray, Jr., "Distortion performance of vector quantization for LPC voice coding," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-30, No. 2, pp. 294-304, April 1982.
- [75] B.H. Juang, "On hidden Markov models and dynamic time warping for speech recognition — a unified view," *AT&T Bell Lab. Tech. J.*, Vol. 63, No. 7, pp. 1213-1243, September 1984.
- [76] B.H. Juang and L.R. Rabiner, "A probabilistic distance measure for hidden Markov models," *AT&T Technical Journal*, Vol. 64, No. 2, pp. 391-408, February 1985.

- [77] —, "Mixture autoregressive hidden Markov models for speech signals," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-33, No. 6, pp. 1404–1413, December 1985.
- [78] B.H. Juang, "Maximum-likelihood estimation for mixture multivariate stochastic observations of Markov chains," *AT&T Technical Journal*, Vol. 64, No. 6, pp. 1235–1249, July-August 1985.
- [79] B.H. Juang, L.R. Rabiner, S.E. Levinson, and M.M. Sondhi, "Recent developments in the application of hidden Markov models to speaker-independent isolated word recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 9–12, 1985.
- [80] B.H. Juang and L.R. Rabiner, "Mixture autoregressive hidden Markov models for speaker independent isolated word recognition," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 1, pp. 41–44, 1986.
- [81] G.E. Kopec, "Formant tracking using hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 3, pp. 1113–1116, 1985.
- [82] —, "A family of formant trackers based on hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 2, pp. 1225–1228, 1986.
- [83] —, "Formant tracking using hidden Markov models and vector quantization," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 4, pp. 709–729, August 1986.
- [84] J.B. Kruskal, "Multidimensional Scaling," Beverly Hills: Sage press, 1978.
- [85] S.E. Levinson, L.R. Rabiner, and M.M. Sondhi, "Speaker independent isolated digit recognition using hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 1049–1052, 1983.
- [86] —, "An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition," *Bell System Technical Journal*, Vol. 62, No. 4, pp. 1035–1074, April 1983.
- [87] S.E. Levinson, "Structural methods in automatic speech recognition," *Proceedings of the IEEE*, Vol. 73, No. 11, pp. 1625–1650, November 1985.
- [88] —, "Continuously variable duration hidden Markov models for speech analysis," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 2, pp. 1241–1244, 1986.
- [89] —, "Continuous speech recognition by means of acoustic/phonetic classification obtained from a hidden Markov model," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 1, pp. 93–96, 1987.

- [90] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, Vol. COM-28, pp. 84-95, January 1980.
- [91] L.R. Liporace, "Maximum likelihood estimation for multivariate observations of Markov sources," *IEEE Trans. Information Theory*, Vol. IT-28, No. 5, pp. 729-734, September 1982.
- [92] R.P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Magazine*, Vol. 4, No. 2, pp. 4-22, April 1987.
- [93] A. Ljolje and F. Fallside, "Synthesis of natural sounding pitch contours in isolated utterances using hidden Markov models," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 5, pp. 1074-1080, October 1986.
- [94] M. Mahmoudian, "La Linguistique," *Editions Seghers*, Paris, 1982.
- [95] J. Makhoul, "Linear prediction: a tutorial review," *Proc. IEEE*, Vol. 63, No. 4, pp. 561-580, April 1975.
- [96] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proceedings of the IEEE*, Vol. 73, No. 11, pp. 1551-1588, November 1985.
- [97] B. Malmberg, "La Phonétique," *Presses Universitaires de France*, Vol. 637, Paris, 1984.
- [98] J.D. Markel and A.H. Gray, Jr., "Linear Prediction of Speech," *Springer-Verlag*, New York, 1976.
- [99] J.L. McClelland and D.E. Rumelhart, "Parallel Distributed Processing. Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models," *MIT Press*, Cambridge, Massachusetts, 1986.
- [100] B. Merialdo, A.M. Derouault, and S. Soudoplatoff, "Phoneme classification using Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 4, pp. 2759-2762, 1986.
- [101] B. Merialdo, "Speech recognition with very large size dictionary," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 1, pp. 364-367, 1987.
- [102] A.M. Michelson and A.H. Levesque, "Error-Control Techniques for Digital Communication," *John Wiley and Sons*, 1985.
- [103] E. Morin, "La Méthode 3. La Connaissance de la Connaissance/1," *Editions du Seuil*, Paris, Juin 1986.

- [104] R. Nag, K.H. Wong, and F. Fallside, "Script recognition using hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 3, pp. 2071-2074, 1986.
- [105] R. Nag, S.C. Austin, and F. Fallside, "Using hidden Markov models to define linguistic units," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 3, pp. 2239-2242, 1986.
- [106] V.V. Nalimov, "In the Labyrinths of Language: a Mathematician's Journey," *ISI Press*, Philadelphia, 1981.
- [107] M. Nishimura and K. Toshioka, "HMM-based speech recognition using multi-dimensional multi-labeling," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 1163-1166, 1987.
- [108] N. Nocerino, F.K. Soong, L.R. Rabiner, and D.H. Klatt, "Comparative study of several distortion measures for speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 25-28, 1985.
- [109] J.W. Oller, Jr., "Coding Information in Natural Languages," *Mouton Co., Printers, The Hague*, 1971.
- [110] A. Papoulis, "Probability, Random Variables, and Stochastic Processes," *McGraw-Hill Book Company*, New York, 1965.
- [111] D.B. Paul, "Training of HMM recognizers by simulated annealing," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 13-16, 1985.
- [112] —, "A speaker-stress resistant HMM isolated word recognizer," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 713-716, 1987.
- [113] J. Perrot, "La Linguistique," *Presses Universitaires de France*, Vol. 570, Paris, 1986.
- [114] A.B. Poritz, "Linear predictive hidden Markov models and the speech signal," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Paris, Vol. 2, pp. 1291-1294, 1982.
- [115] A.B. Poritz and A.G. Richter, "On hidden Markov models in isolated word recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 1, pp. 705-708, 1986.
- [116] L.R. Rabiner and R.W. Schafer, "Digital Processing of Speech Signals," *Prentice-Hall, Inc.*, Englewood Cliffs, New Jersey, 1978.

- [117] L.R. Rabiner and S.E. Levinson, "Isolated and connected word recognition — theory and selected applications," *IEEE Trans. on Communications*, Vol. COM-29, No. 5, pp. 621–659, May 1981.
- [118] L.R. Rabiner, S.E. Levinson, and M.M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent isolated word recognition," *Bell System Technical Journal*, Vol. 62, No. 4, pp. 1075–1105, April 1983.
- [119] —, "On the use of hidden Markov models for speaker independent recognition of isolated words from a medium-size vocabulary," *AT&T Bell Lab. Tech. J.*, Vol. 63, No. 4, pp. 627–642, April 1984.
- [120] L.R. Rabiner, M.M. Sondhi, and S.E. Levinson, "A vector quantizer combining energy and LPC parameters and its application to isolated word recognition," *AT&T Bell Lab. Tech. J.*, Vol. 63, No. 5, pp. 721–735, May-June 1984.
- [121] L.R. Rabiner and S.E. Levinson, "A speaker independent syntax directed connected word recognition system based on hidden Markov models and level building," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-33, No. 3, pp. 561–573, June 1985.
- [122] L.R. Rabiner, B.H. Juang, S.E. Levinson, and M.M. Sondhi, "Recognition of isolated digits using hidden Markov models with continuous mixture densities," *AT&T Technical Journal*, Vol. 64, No. 6, pp. 1211–1234, July-August 1985.
- [123] —, "Some properties of continuous hidden Markov model representations," *AT&T Technical Journal*, Vol. 64, No. 6, pp. 1251–1270, July-August 1985.
- [124] L.R. Rabiner and B.H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, pp. 4–16, Vol. 3, No. 1, January 1986.
- [125] J. Raviv, "Decision making in Markov chains applied to the problem of pattern recognition," *IEEE Trans. Information Theory*, Vol. IT-3, No. 4, October 1967.
- [126] D.R. Reddy, "Speech Recognition", *Academic Press*, New York, 1975; in particular: J.K. Baker, "Stochastic modeling for automatic speech understanding," pp. 521–542.
- [127] E. Rich, "Artificial Intelligence," *McGraw-Hill Book Company*, New York, 1983.
- [128] S. Roucos, J. Makhoul, and R. Schwartz, "A variable-order Markov chain for coding of speech spectra," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Paris, Vol. 1, pp. 582–585, 1982.

- [129] S. Roucos, R. Schwartz, and J. Makhoul, "Segment quantization for very-low-rate speech coding," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Paris, Vol. 3, pp. 1565-1569, May 1982.
- [130] S. Roucos and A.M. Wilgus, "The waveform segment vocoder: a new approach for very-low-rate speech coding," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 236-239, 1985.
- [131] D.E. Rumelhart and J.L. McClelland, "Parallel Distributed Processing. Explorations in the Microstructure of Cognition. Volume 1: Foundations," *MIT Press*, Cambridge, Massachusetts, 1986.
- [132] M.J. Russell and R.K. Moore, "Explicit modelling of state occupancy in hidden Markov models for automatic speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 5-8, 1985.
- [133] M.J. Russell and A.E. Cook, "Experimental evaluation of duration modelling techniques for automatic speech recognition," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 4, pp. 2376-2379, 1987.
- [134] R. Schwartz, Y. Chow, S. Roucos, M. Krasner, and J. Makhoul, "Improved hidden Markov modeling of phonemes for continuous speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 35.6.1-35.6.4, 1984.
- [135] R. Schwartz, Y. Chow, O. Kimball, S. Roucos, M. Krasner, and J. Makhoul, "Context-dependent modeling for acoustic-phonetic recognition of continuous speech," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 3, pp. 1205-1208, 1985.
- [136] R. Schwartz, Y.L. Chow, and F. Kubala, "Rapid speaker adaptation using a probabilistic spectral mapping," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 633-636, 1987.
- [137] S. Singh and K.S. Singh, "Phonetics, Principles and Practice," *University Park Press*, Baltimore, 1976.
- [138] S. Soudoplatoff, "Markov modeling of continuous parameters in speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 1, pp. 45-48, 1987.
- [139] K.N. Stevens, "Acoustic correlates of some phonetic categories," *Journal of the Acoustical Society of America*, Vol. 68, No. 3, pp. 836-842, September 1980.
- [140] K. Sugawara, M. Nishimura, K. Toshioka, M. Okochi, and T. Kaneko, "Isolated word recognition using hidden Markov models," *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, Vol. 1, pp. 1-4, 1985.

- [141] K. Sugawara, M. Nishimura, and A. Kuroda, "Speaker adaptation for a hidden Markov model," *Proc Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 4, pp. 2667-2670, 1986.
- [142] C.C. Tappert, N.R. Dixon, and A.S. Rabinowitz, "Application of sequential decoding for converting phonetic to graphic representation in automatic recognition of continuous speech (ARCS)," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-21, No. 3, June 1973.
- [143] A. Tassy and L. Miclet, "Speaker-independent French digit recognition using word-based vector quantization and hidden Markov models," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 4, pp. 2683-2686, 1986.
- [144] H.P. Tseng, M.J. Sabin, and E.A. Lee, "Fuzzy vector quantization applied to hidden Markov modeling," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 2, pp. 641-644, 1987.
- [145] A.J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Information Theory*, Vol. IT-13, pp. 260-269, 1967.
- [146] A.J. Viterbi and J.K. Omura, "Principles of Digital Communication and Coding," *McGraw-Hill Book Company*, New York, 1979.
- [147] W.D. Voiers, "Methods of predicting user acceptance of voice communication systems," *Final Report to the DCA*, DCA-100-74-C-0056, July 15, 1976.
- [148] C.J. Wellekens, "Explicit time correlation in hidden Markov models for speech recognition," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, Vol. 1, pp. 384-386, 1987.
- [149] R. Whiting and E. Pickett, "A sequential form of Baum's algorithm," *IEEE Trans. Acoustics, Speech, and Signal Processing*, to appear, (see *IEEE Trans. ASSP*, Vol. ASSP-35, No. 5, p. 714, May 1987).
- [150] B.J. Winer, "Statistical Principles in Experimental Design," *McGraw-Hill Book Company*, New York, 1962.
- [151] D.Y. Wong, "Vector/matrix quantization for narrow-bandwidth digital speech compression," *Signal Technology, Inc.*, Report RADC-TR-82-246, 76 pages, September 1982.

VITA

Eric Farges was born on April 18, 1958 in Oran (Algeria) from French parents. Three years later, at his first sign of independence, he decided to cross the Mediterranean Sea to settle down in the beautiful South of France. Several years later he moved to the "Garden of France" (the Loire Valley) where he graduated from high school (lycée Grandmont, Tours) and obtained his Baccalauréat scientific section C with high honor in July 1976. He was then admitted to the "classes préparatoires: mathématiques supérieures et spéciales" at the lycée Descartes (Tours). Being admitted in 1979 to the "Ecole Centrale" in Lyon, he graduated in June 1982 and received the Diploma of Engineer. He also graduated from the University of Saint-Etienne and received a pre-doctoral degree (Diplôme d'Etudes Approfondies) in microelectronics in September 1982. Then came the time to cross the Atlantic Ocean. He headed towards Atlanta and the Georgia Institute of Technology from which he received the M.S. and Ph.D. both in Electrical Engineering in 1983 and 1987 respectively. He held summer job positions as a worker on assembly lines manufacturing electronic components at Radiotechnique-Compelec in 1980, as an engineer developing the control of machine-tools by laser technology at the "Commissariat à l'Energie Atomique (CEA)" in 1981, and as a researcher in GaAs microelectronics with the "Equipe associée au CNRS N° 661" in 1982. During his stay at Georgia Tech he was a 1/3 time graduate teaching assistant for six months, then a 1/2 time graduate research assistant. He also fulfilled his French military service duties for 16 months at Georgia Tech as a "coopérant scientifique." He has been on several sports teams such as judo, ski, volleyball, and European handball. Nowadays he is more into swimming, squash, racquetball and tennis (he's still wondering if it's not too late to make it to the French Open!).

Georgia Institute of Technology

College of Engineering
School of Electrical Engineering
Digital Signal Processing Laboratory
Atlanta, Georgia 30332



GEORGIA TECH 1885-1985

DESIGNING TOMORROW TODAY

December 14, 1987

Mr. Dave Kemp
NSA
Mail Stop R566
Ft. Meade, Maryland 20755

Dear Mr. Kemp:

Please find attached the progress report for Summer Quarter, 1987, for Research Contract MDA 904-85-K-0005: "Research in Digital Speech Processing."

Sincerely yours,

Mark A. Clements
Assistant Professor

MAC:svs

Attachment

Research in Digital Speech Processing

Progress Report

July - September 1987

Abstract

Three major components of the project to date have been:

1. Investigation of improved enhancement techniques,
2. Recognition of components of speech, and
3. Application of Hidden Markov Modeling.

1 Enhancement

The enhancement algorithms previously reported on were moved onto a different computer for better response time in the evaluation process. Experiments were begun on automatic recognition enhancement using hidden Markov Models. Please refer to the attached ICASSP abstract which was submitted and accepted.

2 Recognition of Components of Speech

Work is continuing on the noise modeling ability of Kalman filtering for recognition in a constant background. Currently, we have been unsuccessful in significantly improving performance in SNR's worse than 30dB. Keep in mind, however, that since the vocabulary consists of words which are the same except for interior consonants, an overall level of 30dB SNR puts the portions by which the portions differ only at 0 to 10dB SNR. New methods are being explored.

3 Application of Hidden Markov Modeling

Please find attached a detailed summary of work done in global HMM's. It is excerpted from the Ph.D. proposal of Mr. Eric Farges.

September 1987

Contract MDA 904-85-K-0005

	Budget	Expended	Balance
Personal Services	51,010	38,383	12,628
Fringe Benefits	3,519	4,312	(793)
Materials and Supplies	2,500	1,055	1,445
Travel	3,500	1,498	2,002
Total Direct	60,529	45,248	15,281
Overhead	38,435	28,400	10,035
TOTAL	98,964	73,648	25,316

Constrained Iterative Speech Enhancement With Application To Automatic Recognition

John H.L. Hansen and Mark A. Clements

School of Electrical Engineering,

Georgia Institute of Technology,

Atlanta, GA 30332

A well known speech enhancement technique originally developed by Lim and Oppenheim [1] solves for the maximum likelihood estimate of a speech waveform in additive noise using the constraint that the signal is an all-pole process. Crucial to the success of this approach is the accuracy of the estimates of the all-pole speech parameters at each iteration. Although the original sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, heuristic convergence criteria had to be employed in practice. This restriction makes the approach difficult to apply in environments where automatic enhancement is necessary. In an earlier investigation [2], it was noted that as additional iterations are performed, individual formants of the speech decrease in bandwidth, resulting in unnatural sounding speech. Frame-to-frame pole jitter was also observed, since no explicit frame-to-frame constraints are employed. In ICASSP-87 we introduced a set of iterative speech enhancement algorithms which employed spectral constraints [3]. These algorithms basically employ the Lim-Oppenheim procedure with inter-frame (across time) and/or intra-frame (across iterations) constraints applied after each iteration. These procedures were shown to significantly improve speech quality (as measured by objective speech quality measures including Itakura-Saito, log-area-ratios, weighted spectral slope) over the

unconstrained algorithm and spectral subtraction techniques for an additive white Gaussian noise distortion. In addition, it was also shown that the new constraint techniques improve the objective quality over all speech sound classes.

This paper extends the previous investigation of our constrained enhancement algorithms. Three contributions have been made in this follow-up study. First, the unconstrained Lim-Oppenheim approach required a heuristic convergence criterion. Specifically, the optimum terminating iteration for Itakura-Saito measure was found to be dependent on sound class concentration (see Table 1), and somewhat dependent on SNR. It has been observed that the new constrained enhancement techniques possess a much more consistent terminating criterion over all speech sound classes (see Table 2), and varying SNR's. The best results (through comparison with the known undegraded speech) occur on the same iteration, giving a simple procedure for termination of the algorithm when the undegraded speech is not known (i.e., reality). Second, the constrained techniques have been extended to the colored noise case. Results show that they perform well on non-white, slowly varying aircraft mid-fuselage noise. A comparison with traditional approaches will also be included.

Finally, the new enhancement algorithms are being evaluated as possible preprocessors for automatic speech recognition. The performance of constrained enhancement preprocessing for recognition will be reported over a wide range of SNR for an additive white Gaussian noise distortion. The effectiveness of these constrained approaches will be compared to that of enhancement preprocessing using the unconstrained technique as well as that of spectral subtraction. Two variations are being tested. In the first, all recognition training is performed on the undegraded speech. This serves to model the case of training a recognizer in advance in quiet surroundings (off-line) and using it in a noisy environment. Second, the recognizer training is done using the enhanced speech, which would model training in the field. Early results show marked improvement for recognition performance in a hidden Markov model framework. Detailed results will be reported.

Sound Type	Itakura-Saito Likelihood Measure (across iterations)							
	Original	#1	#2	#3	#4	#5	#6	#7
Silence	1.634	1.615	♣1.608	1.649	1.933	3.756	20.360	49.884
Vowel	4.020	3.721	3.445	♣3.299	3.720	8.319	121.82	—
Nasal	19.814	19.154	18.416	17.656	17.009	16.593	♣15.192	15.697
Stop	7.261	6.114	4.926	3.979	♣3.822	6.889	25.515	29.694
Fricative	3.739	3.637	3.532	♣3.509	3.902	7.658	47.829	94.106
Glide	1.525	1.414	♣1.333	1.442	2.231	4.300	8.391	15.561
Liquid	9.597	8.241	6.546	4.545	2.606	♣1.676	6.381	30.001
Affricate	3.924	3.609	3.213	2.702	2.091	♣1.552	2.911	2.975
Voiced + Unvoiced	5.838	5.321	4.767	4.293	♣4.289	7.346	61.865	—
Total	4.022	3.720	3.402	♣3.151	3.271	5.795	43.457	—

Table 1: Lim-Oppenheim unconstrained speech enhancement for white Gaussian noise. Optimum terminating iteration for best Itakura-Saito distance for a particular speech class (in terms of objective measures) is indicated by a ♣. SNR=+5dB

Sound Type	Itakura-Saito Likelihood Measure (across iterations)								
	Original	#1	#2	#3	#4	#5	#6	#7	#8
Silence	1.634	1.551	1.351	1.155	1.036	0.979	0.929	♣0.884	0.901
Vowel	4.020	3.319	2.865	2.394	1.863	1.677	1.571	♣1.565	1.828
Nasal	19.814	16.490	12.397	10.523	8.682	6.840	4.929	♣3.789	5.548
Stop	7.261	6.246	4.840	3.492	2.668	1.812	1.383	♣1.129	1.435
Fricative	3.739	3.432	3.027	2.612	2.245	1.948	1.729	♣1.615	1.844
Glide	1.525	1.389	1.275	1.232	1.219	1.189	1.161	♣1.153	1.217
Liquid	9.597	6.481	3.382	2.243	1.612	1.209	0.943	♣0.926	1.211
Affricate	3.924	3.722	3.447	3.117	2.806	2.598	2.472	♣2.368	3.966
Voiced + Unvoiced	5.838	4.642	3.658	3.006	2.501	2.131	1.865	♣1.740	1.953
Total	4.022	3.026	2.441	2.069	1.801	1.611	1.457	♣1.381	1.498

Table 2: Hansen-Clements Inter & Intra-frame constrained speech enhancement for white Gaussian noise. Optimum terminating iteration for best Itakura-Saito distance for a particular speech class (in terms of objective measures) is indicated by a ♣. SNR=+5dB

References

- [1] J.S. Lim, A.V. Oppenheim, "All-Pole Modeling of Degraded Speech," *Trans.*

on Acoustics, Speech, and Signal Processing, Vol. ASSP-26, pp. 197-210, June 1978.

- [2] J.H.L. Hansen, M.A. Clements, " Enhancement of Speech Degraded by Non-White Additive Noise," Final Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [3] J.H.L. Hansen, M.A. Clements, " Iterative Speech Enhancement With Spectral Constraints," *Proc. 1987 IEEE ICASSP*, pp. 189-192, Dallas, TX, April 1987.

September 18, 1989

Mr. Dave Kemp
NSA
Mail Stop R556
Ft. Meade, Maryland 20755

Dear Mr. Kemp:

Please find enclosed the Final Report for research contract MDA-904-85-K-0005.

Sincerely yours,

Mark A. Clements
Associate Professor

MAC:kcg

Enclosure

Research in Digital Speech Processing

MDA-904-85-K-0005

Final Report

1 Introduction

Several related research topics were covered in this program. The specific applications are wide-ranging, with the common theme that improved understanding of the basics of speech analysis impacts many areas. The specific areas touched on by this research include very-low-bit-rate coding, phoneme recognition; large hidden Markov model interpretation and implementation; windowless analysis of speech for recognition of transients in noise and quiet; enhancement of speech in additive noise for synthesis and automatic recognition; exploration of alternative front-ends and distance measures for automatic recognition and vector quantization; and enhancement of speech using polyspectra.

2 Research Results

Many details of the research were included in the quarterly progress reports. In the sections below, we will summarize the results and refer to the papers containing more information.

2.1 Large HMM's for Analysis/Synthesis

Please refer to appendices A.1 and A.2, and reference [3].

Continuous speech was examined in the context of constrained analysis/synthesis using large hidden Markov models. In summary, it was determined that an effective decomposition which substantially reduced overall entropy was possible, and that the spectral information of speech could be coded in an intelligible manner down to 125 bits/sec to accomplish this with reasonable computing. Also, the underlying hidden Markov state assumptions were not inconsistent with observed results. All of this was accomplished using discrete observation HMM's where the observations were 10-bit vector quantized codewords. Several natural extensions are suggested from this work: state interpretation, automatic recognition, and generalization to continuous observation large hidden Markov modelling.

2.2 Other Applications of Large HMM's

In an attempt to improve and generalize results, a large HMM was formed using the TIMIT multispeaker data-base. This model had continuous distribution observations based on mel-cepstral and delta-mel-cepstral analysis, the Viterbi and Forward-Backward algorithms. State decoding was performed using the Viterbi algorithm. As summarized in Appendix B.1, 51% speaker independent phoneme recognition was achieved using this method.

2.3 Windowless Techniques for Automatic Recognition

The origin of this work came from the observation that framing boundaries can alter the pattern of performance when an automatic recognizer attempts to distinguish words differing only by short-duration consonants. In this work, methods were developed for training and recognition where different observations as input for every sampling point. One of the front-end analyses involved Kalman filtering, but other methods are also possible. Performance improvement over minimally different words was dramatic; error rate reductions were roughly a factor of 6 (see Appendix C.1). Further work using Kalman filtering extended to noisy input has been performed also with good success (see Appendix C.2).

2.4 Enhancement

Enhancement of speech using iterative Wiener filtering with constraints was implemented and found to improve SNR, subjective quality, and automatic recognition performance. This method was also shown to have good numerical properties. Please see D.1, D.2, and D.3 for details as well as reference [9].

2.5 Recognition of Noisy Speech

Front-end analyses and distance measures have been formulated which are robust in the presence of noise.

The specific one we have examined involved a projection measure of cepstral coefficients and mel-cepstral coefficients. In the context of a continuous-observation hidden Markov model recognizer. This amounts to using a distance of the form:

$$\underline{x}^T C^{-1} \underline{x} - \frac{\underline{x}^T C^{-1} \underline{b} \underline{b}^T C^{-1} \underline{x}}{\underline{b}^T C^{-1} \underline{b}}$$

where \underline{b} is the vector of (mel) cepstral coefficients for the template, \underline{x} is the vector of (mel) cepstral coefficients for the test vector, and C is the pooled covariance matrix for the (mel) cepstral coefficients of the training data. The following results have been obtained.

Type of Coefficients	noise-free		30 dB		20 dB		10 dB		0 dB	
	norm	proj	norm	proj	norm	proj	norm	proj	norm	proj
cepstral	98.5	97.6	98.5	98.2	82.9	96.8	36.8	73.8	7.9	23.8
melcepstral	98.8	98.2	92.9	98.5	65	86.8	22.9	45	15.6	3.8
Normed-melcepstral	98.8	98.5	92.9	98.8	65	89.1	22.9	50	15.6	5

norm = normal

proj = projection

units are in percentages correct

As can be seen, this measure goes a long way in improving performance.

2.6 Polyspectral Techniques for Enhancement

Enhancement using all-pole Wiener filtering is based on speech being all-pole and the noise being something else. Constrained all-pole Wiener filtering improves upon this by incorporating other information concerning the behavior of speech. Use of the bispectrum would further improve matters by allowing separation of Gaussian noise from non-Gaussian speech [11]. The triple correlation of a stationary process is defined by:

$$E(x(n)x(n-i)x(n-j)) = R(i,j).$$

For an all-pole (autoregressive, AR) process, the least squares estimate of the AR parameters can be computed from:

$$\begin{aligned} R\underline{a} &= \underline{\beta} \\ R &= \{r_{ij}\} \quad r_{ij} = R(j-i, j-i) \end{aligned}$$

\underline{a} is the vector of AR parameters:

$$\underline{a} = [1, a, \dots, a_p]^T \text{ and } \underline{\beta} = [\beta, 0, 0, 0, \dots]^T.$$

R is Toeplitz but not symmetric. We have developed a recursion and a lattice filter for solving and representing the system. Although some problems with stability have occurred, the early results were encouraging and will warrant further exploration.

References

1. E. P. Farges and M. A. Clements, "Hidden Markov Models Applied to Very Low Bit Rate Coding," *Proc. of 1986 IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, Tokyo, Japan, April 1986.
2. E. P. Farges and M. A. Clements, "An Analysis/Synthesis Hidden Markov Model of Speech," *Proc. of the 1988 IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, New York, April 1988.
3. E. P. Farges, "An Analysis/Synthesis Hidden Markov Model of Speech," Ph.D. Thesis, Georgia Institute of Technology, School of Electrical Engineering, November 1987, 192 pages.
4. D. J. Pepper and Mark A. Clements, "Large Hidden Markov Model State Interpretation as Applied to Automatic Phonetic Segmentation and Labeling," submitted to *1990 IEEE Int. Conf. on Acoust., Speech, and Signal Processing*.
5. M. A. Clements and S. Lim, "Hidden Markov Model Speech Recognition Based on Kalman Filtering," *Proc. of IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, Dallas, Texas, April 1987.
6. S. Lim and M. A. Clements, "Windowless Analysis of Speech for Automatic Recognition," submitted to *IEEE Trans. on Acoust., Speech, and Signal Processing*.
7. J. H. Hansen and M. A. Clements, "Iterative Speech Enhancement with Spectral Constraints," *Proc. of IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, Dallas, Texas, April 1987.
8. J. H. Hansen and M. A. Clements, "Constrained Iterative Speech Enhancement with Application to Automatic Recognition," *Proc. of the 1988 IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, New York, April 1988.
9. J. H. Hansen and M. A. Clements, "Constrained Iterative Speech Enhancement," submitted to *IEEE Trans. on Acoust., Speech, and Signal Processing*.
10. J. H. L. Hansen, "Analysis and Compensation of Stressed and Noisy Speech with Application to Robust Automatic Recognition," Ph.D. Thesis, Georgia Institute of Technology, School of Electrical Engineering, July 1988, 396 pages.
11. C. L. Nikias and M. R. Raghuveer, "Bispectrum Estimation: A Digital Signal Processing Framework," *Proc. of the IEEE*, vol. 75, no. 7, pp. 869-891, July 1987.

APPENDICES

APPENDIX A

1. Hidden Markov Models Applied to Very Low Bit Rate Speech Coding
2. An Analysis-Synthesis Hidden Markov Model of Speech

APPENDIX B

1. Large Hidden Markov Model State Interpretation as Applied to Automatic Phonetic Segmentation and Labeling

APPENDIX C

1. Hidden Markov Model Speech Recognition Based on Kalman Filtering
2. Windowless Analysis of Speech for Automatic Recognition

APPENDIX D

1. Iterative Speech Enhancement with Spectral Constraints
2. Constrained Iterative Speech Enhancement with Application to Automatic Speech Recognition
3. Constrained Iterative Speech Enhancement

HIDDEN MARKOV MODELS APPLIED TO VERY LOW BIT RATE SPEECH CODING

Eric P. Farges and Mark A. Clements

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia 30332
U.S.A.

Abstract: A new type of very low bit rate speech coder based on a global Discrete Hidden Markov Model (DHMM) of continuous speech for a single speaker is presented here. Several important issues of the training, coding, and decoding procedures are discussed for a 64-state, 1024-observation model. Such a framework is useful in reducing the redundancy in a 10-bit classical Vector Quantizer (VQ), and could lead to a DHMM coder with a bit rate comparable to that of a Segment Vocoder (SV) or a Matrix Quantizer (MQ). This is achieved not only by modelling the long term non-stationarity and the inter-frame time dependencies of the speech, but also by efficiently representing a different kind of information such as vocal tract structure and linguistic patterns.

1. INTRODUCTION

1.1 A new compact and flexible speech model

In recent years, the concept of Hidden Markov Modelling (HMM) of speech has gained considerable popularity due to its successful application in automatic speech recognition. In this paper, we present a new application of such modelling which may be useful to speech coding. In the new model, a 64-state, 1024-observation global Discrete Hidden Markov Model (DHMM) is employed and can be summarized as follows: the system which produces the continuous speech (let us say the human vocal tract) goes through different configurations (called states) as a function of (discrete) time n . A state at time n is a random variable X_n assuming values from a finite state alphabet $S = \{1, 2, \dots, s\}$ ($s=64$). The transitions (or jumps) between the states are probabilistically described by a first order stationary Markov Chain. This in turn is defined by:

- an initial distribution of the states:

$$\pi_0 = (a_i)_{i=1,s} \quad a_i = \text{Prob}(X_1=i) \quad \sum_{i=1}^s a_i = 1$$

- a stochastic transition matrix:

$$\forall n > 1 \quad A = (a_{ij})_{i,j=1,s} \quad a_{ij} = \text{Prob}(X_n=j / X_{n-1}=i) \\ \sum_{j=1}^s a_{ij} = 1$$

This research was supported by grants from the National Security Agency and the Jet Propulsion Laboratory.

The states (like vocal tract articulatory configurations or linguistic patterns) are not directly observable but are hidden. Nevertheless they manifest themselves through "observations" such as LPC spectra of speech segments. In the discrete HMM case the observations Y_n are assumed to be "drawn" from a finite alphabet $O = \{1, 2, \dots, M\}$ ($M=1024$) made of codewords (indices) for a set of template LPC spectra (all pole models) of a classical VQ codebook C . The speech signal is characterized by a sequence of random variables $Y[1:L] = \{Y_1, Y_2, \dots, Y_L\}$.

The production of an observation Y_n at time n is probabilistically governed by the state the system is in at time n . The production rules are described by a stochastic stationary probability output matrix B :

$$B = (b_{ik})_{i=1,s} \quad \forall n > 0 \quad b_{ik} = \text{Prob}(Y_n=k / X_n=i) \\ \sum_{k=1}^M b_{ik} = 1$$

The sequence of random variables $Y[1:L]$ produced by the underlying state sequence $X[1:L]$ and characterized by the discrete probability mass functions (pmf's) $\{b_i(Y_n)\}$ is called a Probabilistic Function of a Markov Chain. To summarize, a DHMM is uniquely defined by $M = (\pi_0, A, B)$.

1.2 Variations to the basic model

The previous model is referred to as an unconstrained model. A constrained model would force a specific structure on the matrix A . For example if $\forall j > i, a_{ij} = 0$, the model is called *left-to-right*.

A time duration constraint can also be applied to the states. With a DHMM, the probability of staying in state i for N units of time is exponentially decreasing:

$$T(N) = (1 - a_{ii}) a_{ii}^{N-1}$$

which might not be very realistic for speech. A Semi-HMM (SHMM) or jump process introduces a state duration pmf $d_i(n)$ for each state i . Then

$$T(N) = d_i(N)$$

where $d_i(n)$ can be chosen to be some discrete probability mass function. The duration of the system in state i is governed by $d_i(n)$, and the state next visited is governed by the transition matrix A . A SHMM is summarized by (π_0, d, A, B) .

A Continuous HMM (CHMM) uses continuously varying observations not limited to a finite discrete alphabet, but to a continuous support U . Continuous pdf's are used for B :

$$B = (b_i(Y))_{i=1,s} \quad \text{for } Y \in U$$

This paper is only concerned with unconstrained DHMM's of speech.

I.3 Motivations for the model

Physical and linguistic reasons: it seems reasonable to consider that the human vocal tract can be represented by a relatively small number of physical configurations (the states), with each configuration being more prone to produce given types of speech spectra (the observations). If one were to assume that speech is composed of a sequence of phonemes, a good description would need to take into account dependencies and relationships between phonemes. In a similar way, the HMM concept statistically identifies time structures and linguistic patterns of the speech and their inter-relationships. Although the underlying states would not represent phonemes, they would exhibit many of the characteristics one would like to see in phonemes, and would enable us to describe speech by a new set of linguistic units. As a result, speech coding could approach bit rates comparable to those of phonemic vocoders, while obviating many of the difficulties. The ultimate bit rate achievable would highly depend on the amount of structure in the underlying model.

Entropy, structure, and bit rate: the entropy of a HMM is defined by [8]:

$$H = - \sum_{i=1}^s \sum_{j=1}^s \pi_i a_{ij} \log_2 a_{ij}$$

where $\pi = (\pi_i)_{i=1,s}$ is the steady state distribution of the states which can be obtained from the transition matrix A by solving the linear system of equations:

$$\begin{cases} \sum_{i=1}^s (a_{ij} - \delta_{ij}) \pi_i = 0 & j = 1, s-1 \\ \sum_{i=1}^s \pi_i = 1 \\ \delta_{ij} = 1 \text{ if } i=j, 0 \text{ otherwise.} \end{cases}$$

The entropy represents the average number of bits necessary to encode the states, and is in some sense a measure of the degree of structure (order) of the system: $0 \leq H \leq \log_2 s$:
- if $\pi_i = 1$, $a_{ij} = 1$ and $\forall j \neq i \pi_j = 0$, $a_{ij} = 0$, then $H = 0$: the minimum entropy corresponds to a completely ordered signal with a total a priori information and a known structure.

- if $\forall i, j \pi_i = 1/s$, $a_{ij} = 1/s$ then $H = \log_2 s$ ($H=6$ for 64 states): the maximum entropy corresponds to a completely disordered signal with no a priori information and no underlying structure. Experimental results presented later show a strong underlying structure in the speech, as measured by entropy.

II. THE SPEECH CODER

The very low bit rate speech coder can be divided into 3 distinct procedures (see fig. 1-3):

- *training* in which the best DHMM, $M = (\pi_0, A, B)$ is found given a training sequence of continuous speech.

- *coding* in which the optimum state sequence $X[1:L]$ given the model M and the observed speech $Y[1:L]$ is found, producing a sequence of codewords.

- *decoding* in which the optimum speech sequence $\hat{Y}[1:L]$ given the model M and the transmitted state sequence $X[1:L]$ is found.

Training and coding take place at the transmitter, decoding at the receiver. An optimum bit representation of the states c_n would be transmitted to the receiver, with the number of bits required depending on the entropy of the model. The experimental work completed includes the follow-

ing: A database of 15 minutes of continuous speech from a single male speaker was digitized at 8 kHz, and LPC models were generated for the 60000 15-ms frames of speech. A 10 bit VQ codebook (1024 codewords) was generated with the binary-split K-means algorithm of Buzo et al. [2], and each speech frame was vector quantized. A 64-state, 1024-observation DHMM was trained through the forward-backward algorithm (FBA). The speech was coded with a trellis coding scheme and decoded with a trellis decoding procedure.

II.1 At the transmitter

Training: to estimate the parameters of the model M a maximum likelihood approach can be used through an iterative procedure called the forward-backward algorithm (FBA). Other approaches such as maximum entropy method could probably be used too. The first proof of convergence of the FBA was introduced by Baum et al. [1], then generalized to multivariate continuous distributions by Liporace [4], and extended to the case of multivariate mixture densities by Juang [3]. A practical use of the algorithm was demonstrated for isolated word recognition by Rabiner et al. [5] for small HMM's ($s=5, M=64, L=4000$). For HMM's as large as this study employed, ($s=64, M=1024, L=60000$) significant modification of previously reported FBA implementations was required to alleviate numerical problems. The results of this training will be discussed in section III.

Coding: given the model M and the observed sequence $Y[1:L]$ the goal is to find the sequence of states $X[1:L]$ which is most likely to have produced $Y[1:L]$. A maximum likelihood approach was used in which the sequence $x[1:L]$ which maximizes:

$$P_c(Y[1:L]/X[1:L], M) = \sum_{n=1}^L a_{x_{n-1}x_n} b_{x_n}(Y_n) \quad (a_{x_0x_1} = a_{x_1})$$

was selected. If we define the likelihood function:

$$\mathcal{L}_c = -\log_{10} P_c, \text{ i.e.,}$$

$$\mathcal{L}_c = \sum_{n=1}^L l_{c_n} \text{ where } l_{c_n} = -\log_{10}(a_{x_{n-1}x_n}) - \log_{10}(b_{x_n}(Y_n))$$

then maximizing P_c is equivalent to minimizing \mathcal{L}_c . A dynamic programming procedure called trellis coding (or Viterbi algorithm) [6] iteratively minimizes the "length" \mathcal{L}_c of the overall state (node) path through a trellis constituted of successive time layers of 64 nodes. The length of a branch between 2 nodes respectively in layers $n-1$ and n is l_{c_n} . The shortest paths ending in each node at every layer n are called "survivors" (there are at most 64 of them). Iterative extensions of the survivors and backtracking lead to the overall best path $X[1:L]$. Some of the trellis coding issues are:

- selection of the initial node (state): The first state can be picked at random, picked according to the steady state likelihood of the states, or picked to minimize the starting lengths. The influence of the starting node is transient, however, and vanishes after a few frames (i.e., survivors starting with different nodes are identical for $n > 5$).

- time constraint: with no time constraint the algorithm could allow as many different states as there are frames per second. It is reasonable, however, to allow no more than 10 (or possibly 20) different states per second. Such a constraint can easily be included in the trellis algorithm through a "sliding window." This feature also makes the coder more robust with respect to noise.

- backward pruning: a full search through the trellis is

or necessary. The extensions at the layer n can be computed from one of up to BP survivors. If no pruning occurs, $P=64$. If backward pruning is performed, $BP < 64$ (for example, c could keep only the first 40 survivors). A pruning down to $BP=20$ generally does not affect the optimality of the results and speeds up the coding algorithm, although even with $BP=64$, the computations are not burdensome.

- weighting factor α : the likelihood \mathcal{L}_c weights transition and output probabilities equally. Given that the output (observation) sequence is known with certainty, it is reasonable to weight B and A differently using the likelihood:

$$\mathcal{L}_c = \sum_{n=1}^L \mathcal{L}_{c_n} : \mathcal{L}_{c_n} = (\alpha-1) \log_{10}(a_{x_{n-1}x_n}) - \alpha \log_{10}(b_{x_n}(y_n))$$

2. At the receiver

Decoding: given the model M and the state sequence $[1:L]$ we want to find the most likely sequence of observations $Y[1:L]$. A similar approach to coding is taken. We define:

$$P_d(y[1:L] / X[1:L], M) = \sum_{n=1}^L [b_{x_n}(y_n)]^{(1-\alpha)} p^\alpha(y_{n-1} / y_n)$$

where $p(y_{n-1}/y_n)$ is the probability of producing observation y_n given that the previous one was y_{n-1} . One alternative for PC-based models is to use the Itakura-Saito distance measure $D(y_{n-1}/y_n)$ for $\log_{10}p(y_{n-1}/y_n)$, giving a likelihood function:

$$\mathcal{L}_d = \sum_{n=1}^L \mathcal{L}_{d_n} : \mathcal{L}_{d_n} = (\alpha-1) \log_{10}[b_{x_n}(y_n)] - \alpha D(y_{n-1} / y_n)$$

here, $D(y_{n-1} / y_n)$ is the LPC log-likelihood ratio.

The likelihood is a compromise between most probable observations in a given state ($\alpha=0$) and maximum smoothness of the LPC spectrum ($\alpha=1$). D is referred to as the smoothness function. It can be computed "on line" at the receiver or a distance matrix (stored at the receiver) can be computed at the transmitter. Storage considerations and decoding computation time should indicate what option to use. If no smoothness function is used, decoding is equivalent to selecting the most probable observation in each state, making the coder look like a 64 level VQ (the codewords of the DHMM being more efficient because of increased structure). Some important issues of the trellis decoding algorithm are:

- initial node: the initial node (observation) is known at the transmitter and is transmitted to the receiver, then the decoding can proceed. However, the quality of the decoded speech decreases when time increases since the very long term (200 frames) predictability of the speech is limited.

- observation pegging: to solve the above problem the actual VQ codeword was transmitted to the receiver every P frames (P is the pegging period as well as the trellis depth and decoding delay). A period $P > 20$ has only a small influence on the overall bit rate. In the limiting case, $P=1$, the DHMM coder is equivalent to a 1024 level VQ.

- backward pruning: same as coding with $BP=100$.

- forward pruning: to decrease the decoding time without affecting the optimality of the results, at time n the algorithm looks ahead and selects only the FP most probable observations in state X_{n+1} for which extensions should be computed (need $FP \geq BP$, use $FP=100$).

- weighting factor α : the initial implementation used a constant α . Results indicate that α should be variable:

. in steady state speech regions the smoothness function should be weighted more ($0.5 < \alpha \leq 1$).

. in transient speech regions the output matrix B should

be made adaptable with time $\alpha = \alpha_n$ based on the evolution of the state sequence.

III. PRELIMINARY EXPERIMENTAL RESULTS

The training procedure was run until convergence was approached, with a likelihood value of

$$\log_{10} P = -3.352 \times 10^5$$

Although the model was still improving slightly, the main structures of the matrices were already there (see figures 4-6):

- the initial distribution of the states converged quickly to $a_{19}=1$ and for $i \neq 19$ $a_i=0$.

- the A matrix displayed, as expected, a strong diagonal structure (see figure 6).

- 56.4% of the entries in A and 89.3% in B were less than 10^{-3} (see figures 4,5).

For a classical VQ, a (non hidden) Markov model was generated by defining the codewords as the states (i.e., states were observations) and a transition matrix was computed by frequency counts on the 15 minutes of speech. The resulting entropy for a 6-bit codebook (64 codewords) was found to be $H = 3.9$. The transition matrix of the 64 state HMM computed on the same data had an entropy of 2.6, which suggests that an inherent underlying structure exists in speech which is not taken into account by the 6-bit VQ. Such techniques as Segment Vocoding and Matrix Quantization capture part of this structure but not all [7].

Preliminary experiments, using the simplest of the coding-decoding techniques described, produced speech superior in quality to that of a 6-bit vector quantizer, but inferior to that of a 10-bit VQ. The work, at this point in time, is very encouraging, considering the large number of unexplored possibilities available.

Conclusion:

A statistical derivation of a new type of speech model has been presented: a global 64 state, 1024 observation DHMM of continuous speech has been proven to be practical, compact, and general, but flexible, automatically trainable, and bit rate efficient (as low as 2.6 bits / frame).

A complex but important underlying structure has been brought to light. Although initially speaker dependent, this new speech model could become speaker independent when plausible linguistic units represented by the states and derived from the strong structure of the model are identified.

Refinements in the modelling and coding-decoding process should produce good speech quality for a very low bit rate coder. Among the many other applications of a DHMM of speech, the detected underlying state sequence could be used for continuous speech recognition.

References:

- [1] L.E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, 41, pp. 164-71, 1970.
- [2] A. Buzo, A.H. Gray, R.M. Gray, J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-28, No.5, pp.562-74, October 1980.
- [3] B.H. Juang, "Maximum-likelihood estimation for mixture multivariate stochastic observations of Markov chains," *AT&T Technical Journal*, 64, No.6, July-August 1985.

[4] L.R. Liporace, "Maximum likelihood estimation for multivariate observations of Markov sources," *IEEE Trans. Information Theory*, IT-28, No.5, pp.729-34, September 1982.

[5] L.R. Rabiner, S.E. Levinson, and M.M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent, isolated word recognition," *Bell System Tech. J.*, 62, No.4, pp.1075-1105, April 1983.

[6] G.D. Forney, JR., "The Viterbi Algorithm," *Proc. IEEE*, 63, No.3, pp. 268-78, March 1973.

[7] D.Y. Wong, "Vector/Matrix quantization for narrow bandwidth digital speech compression," *Signal Technology Technical Report RADC-TR-82-246*, September 1982.

[8] S. Roucos, J. Makhoul, R. Schwartz, "A variable-order Markov chain for coding of speech spectra," *Proc. ICASSP*, Paris, pp.582-5, 1982.

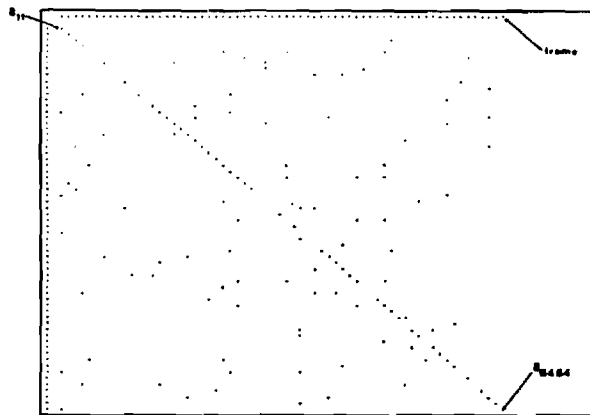


figure 6: Entries (dots) in transition matrix A greater than 0.1 .

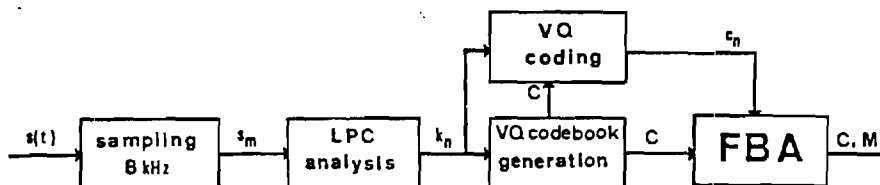


figure 1: Training of the DHMM at the transmitter.

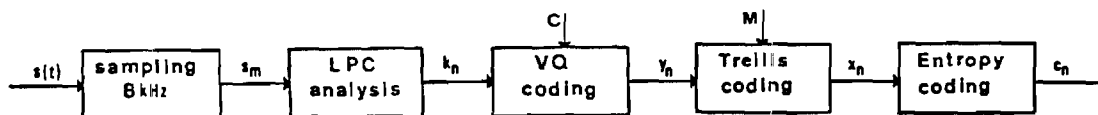


figure 2: DHMM coding at the transmitter.

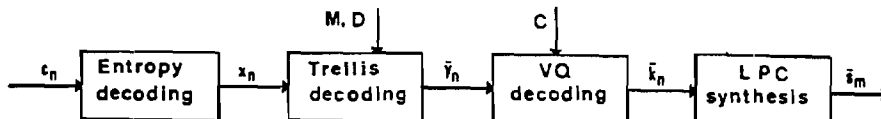


figure 3: DHMM decoding at the receiver.

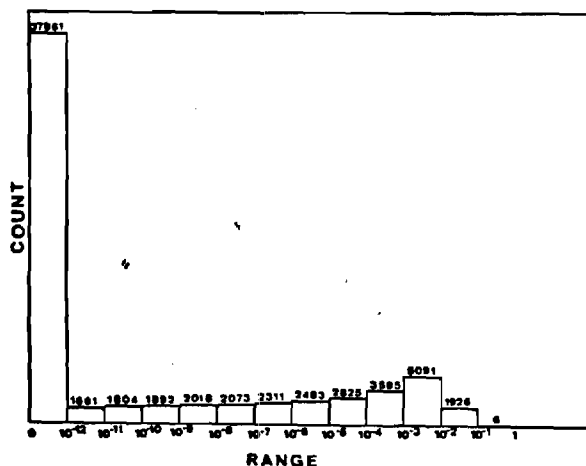


Figure 4: Histogram of entries of output probability matrix B.

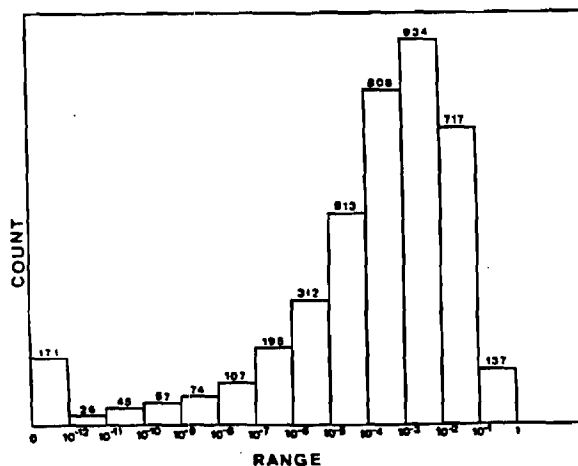


figure 5: Histogram of entries of transition matrix A.

9. 1. 4

AN ANALYSIS-SYNTHESIS HIDDEN MARKOV MODEL OF SPEECH

Eric P. Farges¹ and Mark A. Clements

Georgia Institute of Technology
School of Electrical Engineering
ATLANTA, GA 30332

ABSTRACT

Continuous speech from one male American speaker is described by a single large hidden Markov model (HMM). Problems of model parameter estimation, speech decoding, and speech-observation decoding are presented under a unified conceptual framework built around a maximum likelihood approach. The model has 64 states and 1024 observations, and is assumed to be fully connected. This model possesses not only speech analysis but also speech synthesis capabilities. Very low bit rate speech coding is made possible by the fact that very little information is needed to encode speech state sequences. Intelligible speech, comparable in subjective quality to 10 bits/frame vector quantized speech, can be constructed from the state sequences with bit rates as low as 5 bits/frame (to encode the LPC spectral information). Intelligible speech was produced at 1.68 bits/frame, whereas a 2 bits/frame pitch excited LPC vector quantization did not produce intelligible speech.

1 Introduction

This paper is a brief summary of the research we started in 1983 at Georgia Tech in the DSP group of the School of Electrical Engineering. A detailed account of this research can be found in Farges [3,2,1] and in Farges et al. [4,5,6,7,8]. The objective was to develop a single large HMM of continuous speech which would provide both analysis capabilities in terms of state sequences and speech synthesis (reconstruction) capabilities from state sequences, leading to such applications as very-low-bit-rate speech coding and continuous speech recognition. Our approach departs from "classical" HMM's representations because:

- we use a single large HMM (64 states, 1024 observations) directly applied to continuous speech as opposed to several small HMM's (5 to 10 states, 64 to 200 observations) applied to "segmented" parts of speech (such as words, phonemes, etc.) and then concatenated together.

¹This author is now with:
Standard Elektrik Lorenz AG
Postfach 10
7030 Stuttgart 40
Germany

- We use a large database of continuous speech (15 minutes or more) to perform a totally automatic and computerized model parameter estimation (no segmentation in terms of basic "speech units" is needed).

- We introduce and develop the concept of speech synthesis (reconstruction) from a state sequence to produce quality speech at very low bit rates.

- We develop a unified conceptual framework, as well as practical efficient algorithms, to solve the class of problems related to large HMM's.

- We introduce several new types of speech coders and describe the achievable bit rates with the evaluation of the associated subjective speech quality.

- We describe some encouraging results about the "physical meaning" of the states and their "statistical interpretation" in terms of classical linguistic units.

The basic implementation of such an HMM of speech is summarized in the block diagrams of figure 1.

2 Model estimation

The forward backward algorithm (FBA) was adapted and optimized to accommodate large training databases. An accurate and efficient scaling procedure was developed on top of the FBA to eliminate underflow problems [3,5]. The optimization process is summarized in table 1. The maximum likelihood (ML) estimation process proved experimentally to be also a minimum entropy process (see figures 2 and 3).

3 State decoding

The state decoding was performed by an unconstrained trellis decoding algorithm (TDA) operating on the state-space trellis of the HMM. The concept of convergence nodes was introduced. A recursive partial backtracking algorithm was developed to detect them. It allowed to optimally decode small "blocks" of speech independently of past and future blocks — therefore decreasing the memory requirements of the algorithm. The distribution of the state-decoder delay is shown in figure 4 (1 time unit = 15ms).

4 Observation decoding

Several observation decoding schemes were defined and experimented with. An adjoined model $\hat{\lambda}$ and an inverse model λ^{-1} were defined. The observation decoding was performed by a constrained trellis decoding algorithm operating on the observation-space trellis of the adjoined or inverse HMM. The general unified ML framework of HMM's is summarized in table 5.

5 Applications to very-low-bit-rate speech coding

Several new types of speech coders were introduced: the partitioned HMM-VQ and the HMM-VQ including the "most probable" coder, the "expected" coder, and the constrained TDA coder. It was shown that HMM coders can achieve lower bit rates (for the same speech quality) than simple Markov-VQ's (see tables 2,3, and 4). Quite intelligible speech was synthesized with a 1.68 bits/frame bit rate — a little less than the 2 bits/frame bit rate of a 4-level VQ which did not produce intelligible speech with the same pitch excitation data. Speech synthesized from bit rates of 4.68, 3.68, 2.88, and 2.54 bits/frame were judged indistinguishable from the speech synthesized by the classical 10 bits/frame vector quantizer (see subjective test results of table 4).

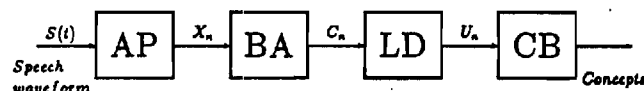
6 Conclusion

It has been demonstrated that it is feasible to generate a large and global HMM of speech. The model captures a significant part of the underlying language structure of the speech waveform. Therefore it enabled us to encode and synthesize speech efficiently at very low bit rates.

References

- [1] E.P. Farges, "Hidden Markov models for speech processing," *Ph.D. Qualifier Report*, Georgia Institute of Technology, School of Electrical Engineering, May 1984, 30 pages, (unpublished).
- [2] —, "An analysis-synthesis hidden Markov model of speech," *Ph.D. Dissertation Proposal Report*, Georgia Institute of Technology, School of Electrical Engineering, May 1987, 113 pages, (unpublished).
- [3] —, "An analysis-synthesis hidden Markov model of speech," *Ph.D. Thesis*, Georgia Institute of Technology, School of Electrical Engineering, November 1987, 192 pages.
- [4] E.P. Farges and M.A. Clements, "Hidden Markov models applied to very low bit rate speech coding," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tokyo, Vol. 1, pp. 433-436, 1986.

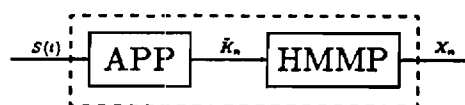
- [5] —, "Practical estimation of the parameters of large hidden Markov models," *IEEE Trans. on Information Theory*, 1988, to appear.
- [6] —, "An efficient trellis decoding algorithm," *IEEE Trans. on Communications*, 1988, to appear.
- [7] —, "An analysis-synthesis hidden Markov model of speech: a unified approach," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 1988, to appear.
- [8] —, "Speech analysis by hidden Markov models," *Journal of the Acoustical Society of America*, 1988, to appear.



a) Speech processing unit: the DSP implementation of the listener side of the message model of human communication.

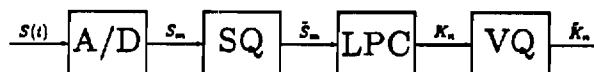
AP : Acoustic Processor
BA : Bit Allocator
LD : Linguistic Decoder
CB : Concept Builder

X_n : Acoustic patterns
 C_n : Acoustic codewords
 U_n : Linguistic units



b) The acoustic processor.

APP : Acoustic Pre-Processor
HMMP : Hidden Markov Model Processor



c) The acoustic pre-processor.

A/D : Analog to Digital converter
SQ : Scalar Quantization
LPC : Linear Predictive Coding
VQ : Vector Quantization

Figure 1: Basic speech processing unit.

Table 1: The FBA: a long optimization process.

Algorithm type	CPU time†
original maximisation problem	Infeasible
Forward Backward Algorithm (FBA)	theoretically feasible practically Infeasible
scaled FBA	CPU > 30 hours
partially optimised, scaled FBA	20 h < CPU < 30 h
optimised FBA with partial scaling	CPU < 20 h
optimised, log-less FBA with partial scaling	CPU = 1 hour 42 minutes

†For the algorithm operating on the 15min of speech on a Data General MV/10000.

Table 2: Markov-VQ bit rate.

Codebook Size	Bit rate (bits/frame)		Bit rate reduction (%)
	VQ	Markov-VQ	
1024	10	5.3	47
64	6	3.9	35
32	5	3.2	36
16	4	2.4	40
8	3	1.7	43
4	2	1.0	50
2	1	0.4	56

Table 3: Bit rate as a function of the pegging period K ($H_A = 1.68$, $H_B = 5.90$).

K (frames)	2	3	4	5	6	7	8
B(K) (bits/frame)	7.67	4.68	3.68	3.18	2.88	2.68	2.54
K (frames)	9	10	11	12	13	14	15
B(K) (bits/frame)	2.43	2.35	2.28	2.22	2.18	2.14	2.11

Table 4: Results of the subjective quality testing.

Systems: Session 1		Systems: Session 2	
System	Speech/Coder type	System	Speech/Coder type
A1	pitch excited LPC	B1	pitch excited LPC
A2	1024-level VQ	B2	1024-level VQ
A3	HMM "most probable"	B3	HMM $K=4$
A4	HMM "expected"	B4	HMM $K=6$
A5	HMM $K=3$	B5	HMM $K=8$
A6	4-level VQ	B6	4-level VQ

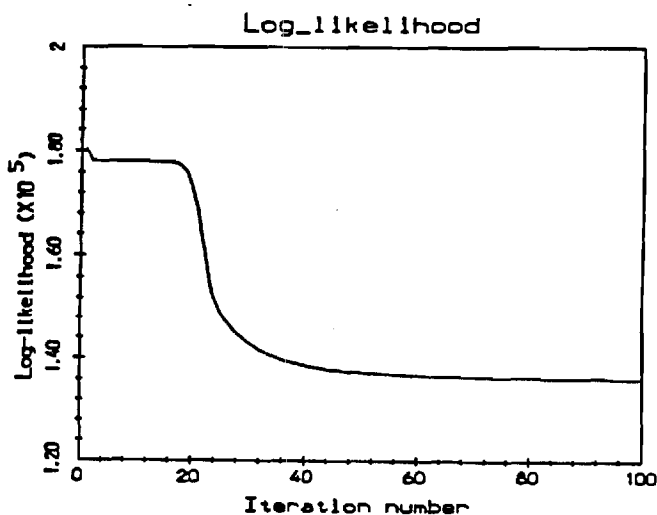


Figure 2: Evolution of the log-likelihood.

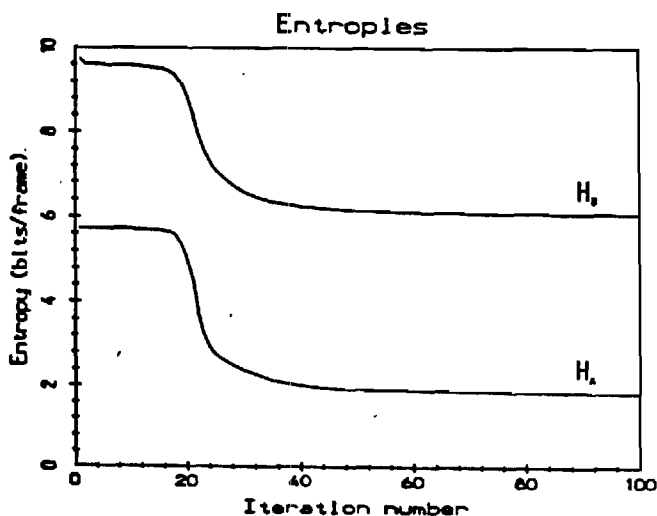


Figure 3: Evolution of the entropies.

Statistics: Session 1				Statistics: Session 2			
Rank	System	Average Score	Standard Deviation	Rank	System	Average Score	Standard Deviation
1	A1	59.9	35	1	B1	58.4	34
2	A2	47.5	28	2	B2	45.2	28
3	A5	38.7	25	3	B3	38.0	24
4	A4	34.8	23	4	B4	37.8	24
5	A3	33.3	23	5	B5	36.3	23
6	A6	21.1	15	6	B6	18.1	12

Significance matrix: Session 1						
Systems	1	2	3	4	5	6
1	-	-	-	-	-	-
2	1	-	-	-	-	-
3	2	0	-	0	0	-
4	2	0	-	-	0	-
5	2	0	-	-	-	-
6	2	2	1	1	1	-

Significance matrix: Session 2						
Systems	1	2	3	4	5	6
1	-	-	-	-	-	-
2	1	-	-	-	-	-
3	2	0	-	-	-	-
4	2	0	0	-	-	-
5	2	0	0	0	-	-
6	2	2	2	2	2	-

Significance Level	Meaning
0	no significant difference
1	significant difference 5% probability of error
2	more significant difference 1% probability of error

†Error: to say that two coded sentences are significantly different when they are not.

Table 5: Analysis-Synthesis HMM's of Speech: summary.

Sub-Optimum Maximum Likelihood Approach	
Phase	Procedure
training (estimate λ)	$\sum_s P_A(z, Y) = \max_{\lambda} \sum_s P_{\lambda}(z, Y)$
state decoding (estimate X)	$P_A(X, Y) = \max_{\lambda} P_{\lambda}(z, Y)$
observation decoding (estimate Y)	$P_A(X, \hat{Y}) = \max_{\lambda} P_{\lambda}(X, y)$ with $\hat{Y}_i = Y_i$, $\hat{Y}_K = Y_f$
Optimum Maximum Likelihood Approach	
Phase	Procedure
training (estimate λ)	$\sum_s P_A(z, Y) = \max_{\lambda} \sum_s P_{\lambda}(z, Y)$
state decoding (estimate X)	$P_A(X, Y) = \max_{\lambda} P_{\lambda}(z, Y)$
training (estimate λ^{-1})	$\sum_y S(y, Y) P_{\lambda^{-1}}(X, y) = \max_{\lambda^{-1}} \sum_y S(y, Y) P_{\lambda^{-1}}(X, y)$
observation decoding (estimate Y)	$P_{\lambda^{-1}}(X, \hat{Y}) = \max_{\lambda^{-1}} P_{\lambda^{-1}}(X, y)$ (with or without $\hat{Y}_i = Y_i$, $\hat{Y}_K = Y_f$)

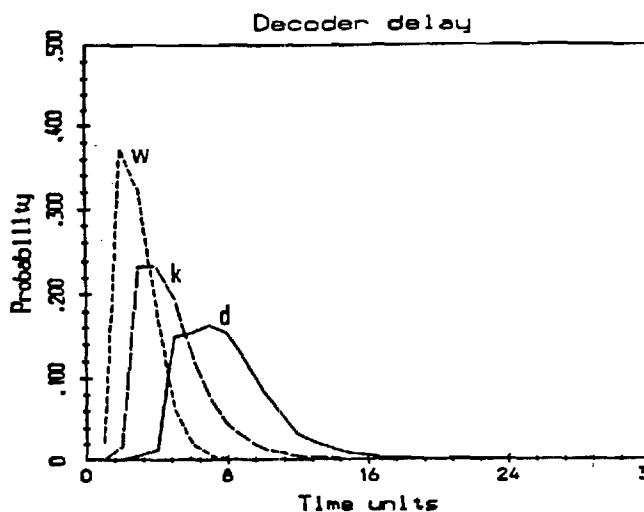


Figure 4: Distributions of the convergence node weight W , convergence length K , and decoder delay d .

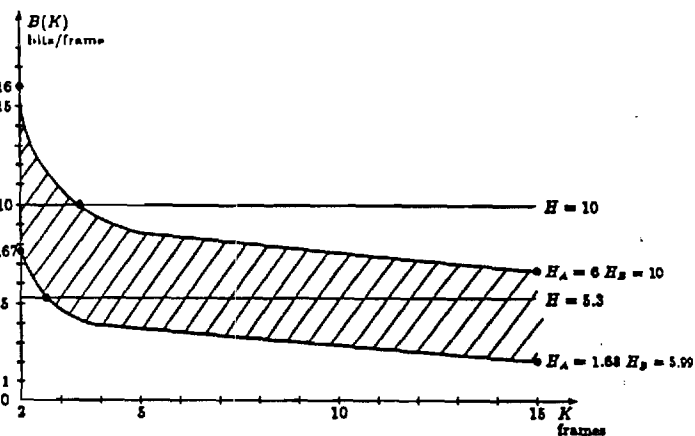


Figure 5: Range of the HMM-VQ bit rate as a function of the pegging period.

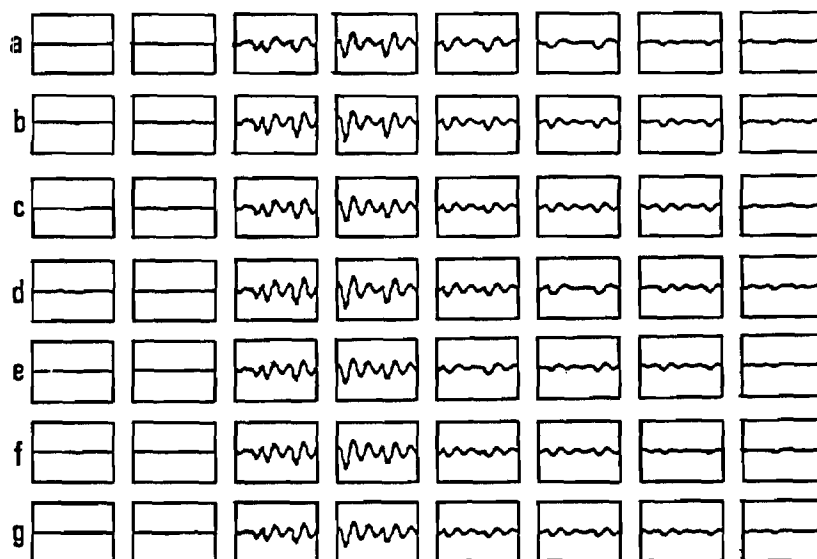


Figure 6: Examples of reconstructed waveforms.

- a) original 1024-level VQ speech
- b) "expected" reconstructed speech
- c) "most probable" reconstructed speech
- d) HMM ($K = 3$) reconstructed speech
- e) HMM ($K = 4$) reconstructed speech
- f) HMM ($K = 6$) reconstructed speech
- g) HMM ($K = 8$) reconstructed speech

Large Hidden Markov Model State Interpretation As Applied To Automatic Phonetic Segmentation And Labeling

David J. Pepper and Mark A. Clements

Hidden Markov models with a large number of states (approximately 64) can be used to model spoken language. These models have proven themselves to be quite useful in low bit rate speech coding applications as well as in automatic phonetic recognition. Due to the statistical nature of the training algorithms (the Forward-Backward and Viterbi algorithms), it is very difficult to determine what the individual model states actually represent. This paper presents one possible interpretation of the model states in terms of the phonetic structures contained within the hidden Markov model. It will be shown that these phonetic structures possess a number of common features which lead to an understanding of how the statistical training algorithms arrange the states of a hidden Markov model. These common features include entry states, plateau states, and exit states; i.e., the initial, central, and transition states of the phoneme, respectively. This paper will also show how these phonetic structures can be used to segment and label the phonemes in an input sentence automatically using only the state sequence produced by the Viterbi decoding algorithm.

These large hidden Markov models are best termed language models, since the resulting systems, which are trained on entire sentences of speech, represent all of the possible acoustic transitions contained within naturally occurring fluent speech. Thus, the resulting systems model between-word coarticulation effects as well as phonetic transitions within each word. These global and local coarticulation effects are thereby accounted for in the phonetic models described above, producing very accurate systems of phonetic models. This work has not yet been extended to include higher level language or word models, but such models can be easily included in these systems to significantly improve the recognition results quoted below.

The large hidden Markov models used in this study were trained on the multi-speaker TIMIT Acoustic-Phonetic database using a continuous observation density Forward-Backward training algorithm with a two stage model initialization. On a 104 speaker subset of the TIMIT database, the phonetic recognizer was able to achieve a 51% recognition rate with 11% insertions for a constrained recognition experiment and a 49% recognition rate with 12% insertions for the unconstrained case. These recognition rates compare very favorably with previous studies of this database, and thus verify the accuracy of the phonetic models developed in this work.

HIDDEN MARKOV MODEL SPEECH RECOGNITION BASED ON KALMAN FILTERING

Mark A. Clements and Sungjae Lim

School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Abstract

Traditional hidden Markov model speech recognition is generally based on a set of parameters (often LPC related) which are extracted at discrete intervals. Such an analysis necessitates use of a discrete-trial hidden Markov model in which the underlying states can only change at intervals related to the frame rate of the analysis. The exact locations of the analysis windows used can influence the front-end outputs and as a result can cause confusion between words differing in short-duration consonants. In the current study, an alternate method which does not require segmentation is proposed, and a simple version is implemented. The discrete trial hidden Markov model algorithms are adapted to this framework leading to significantly improved recognition performance.

Introduction

Over the past few years, the method of choice for many speech recognition applications has been based on hidden Markov modeling. Steady improvement has been reported in such areas as speaker independence, noise handling, training and response times, as well as general performance. The first HMM based systems modeled speech as a discrete state discrete trial Markov process with discrete observations. Recently, new models which allow a continuous distribution of observations have been presented. Although the methods for accomplishing this differ, they all eliminate the vector quantization step and virtually all report improved performance. Throughout all these models, however, the assumption remains that sampling the parameterization of the speech (e.g., spectral or LPC based parameters) is only necessary every 10 to 30 milliseconds. When words differ only by a short duration interior consonant, however, the exact placement of the analysis windows can have an impact on performance. The current study is the result of an attempt to eliminate these windowing effects in an efficient manner. The front-end is based on a Kalman filtering model which produces an output for every sample point. The matching algorithm can be considered an approximation to a discrete state continuous transition hidden Markov modeling technique.

Front-End Analysis

A general autoregressive representation of speech can be based on a model of the form:

$$\mathbf{x}(k) = \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{B}(k)\mathbf{u}(k) + \mathbf{G}(k)\mathbf{w}(k) \quad (1a)$$

$$\mathbf{y}(k) = \mathbf{C}^T(k)\mathbf{x}(k) + \mathbf{v}(k) \quad (1b)$$

where the vector $\mathbf{x}(k) = [x(k), x(k-1), \dots, x(k-p+1)]^T$, $x(k)$ is speech without noise, $u(k)$ is a periodic input and $B(k)$ its gain, $w(k)$ a noise input and $G(k)$ its gain, $C(k)=[1, 0, 0, \dots]$ varying vocal-tract filter. The sequence $y(k)$ is the digitized speech and is the same as $x(k)$ with no observation noise.

Systems similar to this have been used to model many varied signals arising in innumerable applications. In the linear prediction synthesis model $A(k)$ remains constant over 10 to 30 millisecond intervals, one of $u(k)$ or $w(k)$ is usually set to zero, and $v(k)$ is zero. In the LPC analysis model, $u(k)$ and $v(k)$ are generally assumed to be zero so that $A(k)$ can be estimated every 10 to 30 milliseconds. Recursive linear least squares estimation based on the general model falls within the general area of Kalman filtering, which allows one to efficiently compute the least squares estimate of $\mathbf{x}(k)$ from the least squares estimate of $\mathbf{x}(k-1)$ and $y(k)$. The property we wish to exploit is that if we have modeled the system correctly, the prediction error, $v(k)$, would be white. Even if the system model is correct, the prediction error signal $v(k)$ will not be zero due to the noise terms. It should have a predictable ratio of its power to the unfiltered signal's power, however. If there are L possible models from which the observed signal was generated, this idea can be used for computing the relative likelihoods of each model given the observed signal. Denote $v_i(k)$ the prediction residual (innovations sequence) for system i given observations $y(1), y(2), \dots, y(k)$, and $p_i(k)$ the probability of system i given $y(1), y(2), \dots, y(k)$. It has been shown [1] that

$$p_i(k+1) = \frac{N(v_i(k), V_i(k))p_i(k)}{\sum_j N(v_j(k), V_j(k))p_j(k)} \quad (2)$$

where $V_i(k)$ is the variance of $v_i(k)$ if model i is correct, and $N(a,b)$ represents the Gaussian density of mean zero and variance b evaluated at a . Computation of $V_i(k)$ and $v_i(k)$ can be performed recursively using the Kalman filtering equations, with $V_i(k)$ computed off-line. It should be noted that if $v(k)$ and $u(k)$ are set to zero, and $A(k)$ and $G(k)$ are allowed to change only at abrupt intervals, then $v_i(k)$ becomes an LPC residual for model i .

Choosing the value of i which maximizes $p_i(k)$ is in many ways like implementing an LPC vector quantizer. Several important differences exist, however. First, different forms of the models can be used. This would, for example, allow different order models for different sounds. Second, periodic components could specifically be put into some models through $u(k)$. Third, additive colored noise can be modeled explicitly. Fourth, $A(k)$ can be a time varying transition matrix which could be used in a manner similar to matrix quantization in speech coding. And fifth, the probability calculations via equation (2) is not merely an Itakura distance. The above described procedure has been applied with success to cardiac arrhythmia detection [2] and stochastic aircraft control [3].

Despite the potential power of this technique, a number of difficulties remain. Most notable is the training procedure for the models appropriate to speech. Since this research was intended mainly to study the effects of eliminating the windows

in the speech analysis, a greatly simplified model was adopted. The adoption of this simplified form of equations (1) could in no way be construed to mean that more elaborate models could not be trained, but merely that it was deemed more appropriate to use simple models in a first set of experiments. The models used were:

$$\mathbf{x}(k) = \mathbf{A}(k)\mathbf{x}(k-1) + \mathbf{G}\mathbf{w}(k) \quad (3a)$$

$$\mathbf{y}(k) = \mathbf{C}^T \mathbf{x}(k) \quad (3b)$$

where $\mathbf{w}^T(k) = [w(k), w(k-1), \dots, w(0)]$ and the sequence $w(k)$ is white Gaussian noise of zero mean and variance q which is correlated with all values of $x(m)$, $m < k$.

$$\mathbf{A} = \begin{bmatrix} a(1) & a(2) & \dots & a(p) \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} \quad (4)$$

$$\mathbf{C}^T = [1, 0, 0, \dots] \quad (5)$$

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix} \quad (6)$$

$$\mathbf{Q} = \begin{bmatrix} q & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{bmatrix} \quad (7)$$

The Kalman filtering equations become

$$\begin{cases} \mathbf{v}(k) = \mathbf{y}(k) - \mathbf{C}^T \hat{\mathbf{x}}(k|k-1) \end{cases} \quad (8a)$$

$$\begin{cases} \hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{M}(k)\mathbf{v}(k) \end{cases} \quad (8b)$$

$$\begin{cases} \hat{\mathbf{x}}(k+1|k) = \mathbf{A} \hat{\mathbf{x}}(k|k) \end{cases} \quad (8c)$$

$$\begin{cases} \mathbf{V}(k) = \mathbf{C}^T \mathbf{P}(k|k-1) \mathbf{C} \end{cases} \quad (8d)$$

$$\begin{cases} \mathbf{M}(k) = \mathbf{P}(k|k-1) \mathbf{C} \mathbf{V}^{-1}(k) \end{cases} \quad (8e)$$

$$\begin{cases} \mathbf{P}(k|k) = \mathbf{P}(k|k-1) - \mathbf{M}(k) \mathbf{C}^T \mathbf{P}^T(k|k-1) \end{cases} \quad (8g)$$

$$\begin{cases} \mathbf{P}(k+1|k) = \mathbf{A} \mathbf{P}(k|k) \mathbf{A}^T + \mathbf{G} \mathbf{Q} \mathbf{G}^T \end{cases} \quad (8h)$$

where $\hat{\mathbf{x}}(i|j)$ is the best estimate of $\mathbf{x}(i)$ given $\mathbf{y}(0), \mathbf{y}(1), \dots, \mathbf{y}(j)$; and $\mathbf{P}(i|j)$ is the covariance matrix of $\mathbf{x}(i)$ given $\mathbf{y}(0), \mathbf{y}(1), \dots, \mathbf{y}(j)$. If white observation noise of variance Z were included, equation (8d) would merely have a Z added. The last four of these equations are computable off-line. The only training that needs to be done is for the \mathbf{A} matrices, which are all of the same form. It was decided that the values $a(1), a(2), \dots, a(p)$ for all L models could be computed by training a p th order LPC vector quantizer with L codewords. In this study, p was 64 and L was 10. The vector quantizer training was done on speech produced by one talker speaking roughly 40 seconds of continuous speech. Twenty millisecond Hamming windows were applied every 10 milliseconds. Vector quantizer training was accomplished using a binary split algorithm. It may be seen at this point that the use of windows and frames in training is

counter to the goals of the model. However, since so many frames were used in training (40,000), and since such a large number of different frame positions were sampled, a highly representative set of systems was undoubtedly compiled.

The recursion of equations (8) were initialized using

$$\begin{bmatrix} R_0 & R_1 & R_2 & \dots & R_9 \\ R_1 & R_0 & R_1 & \dots & R_8 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R_9 & R_8 & R_7 & \dots & R_0 \end{bmatrix} \quad (9)$$

where R_i is the i th autocorrelation lag.

The variance in equation (2) is linear with respect to the signal power, and should be normalized. This was done using a second order window on the squared signal whose z -transform was

$$\frac{1}{(1-0.961z^{-1})^2} \quad (10)$$

The time constant was roughly 10 milliseconds.

Another issue dealt with how the system should be constrained so that it would not "lock on" to a specific model and make it difficult to track a change of model. Following [2], we set limits on the maximum and minimum values $p_i(k)$ could achieve. A post-analysis of results, however, showed this step to be of little consequence in the current system.

The system described above was implemented and used to analyze some continuously spoken sentences. The output of the analysis consisted of one 6-bit number for every speech sample analyzed (8,000/second). Several observations are worth pointing out at this time. First, long runs of the same codeword (greater than 10 milliseconds) were usually seen in fairly steady-state portions of the speech. However, in transition and many consonant regions, codeword runs could be very short (less than 2 milliseconds). Second, voiced sounds usually produced a string of codewords which were one value for most of the pitch period, and another for the remainder of the period. This behavior seems reasonable since periodic input was not put into any of the models. Third, speech could be synthesized from the strings of codewords by simply inputting noise into equations (3) and allowing the filter to change every sample. Although poor in quality, this speech was superior to that produced by a conventional 6-bit LPC vector quantizer updated every 10 milliseconds.

Hidden Markov Modeling

In some versions of the hidden Markov model speech recognizers, a vector quantizer codeword every 10 to 30 milliseconds is all the recognizer or training procedure sees. The system of the current study can supply a recognizer with a codeword every speech sample and hopefully eliminate windowing artifacts. Unless a method could be formulated which would reduce the computational burden by at least an order of magnitude, however, such an expansion of data would be quite unmanageable. Such a method which exploits the fact that codewords often come in long bursts will be presented below.

Notation

Due to the use of certain symbols in the earlier part of this paper, it is necessary to use some non-standard notation to avoid confusion. For a discrete state discrete transition discrete observation hidden Markov model, we must define:

n = number of states

N_v = number of possible observations (vectors) = 64

Π_i = probability the model starts in state i , and

$$\Pi^T = [\Pi_1, \Pi_2, \dots, \Pi_n]$$

T = transition probability matrix, where :

T_{ij} = probability of transit from state i to state j in one trial; $i=1, \dots, n$; $j=1, \dots, n$.

B = observation probability matrix where

b_{jk} = probability of observing codeword k given state $j = b_j(k)$.

$O(t)$ = codeword observed at time t .

$R(t)$ = observation matrix, consisting of:

$$R(t) = \text{diag}[b_1(O(t)), b_2(O(t)), \dots, b_n(O(t))]$$

The transitions were constrained to be left-to-right making T upper triangular.

For a given model M , and observations $O(1), O(2), \dots, O(F)$, we define

$$\alpha^T(t) = [\alpha_1(t), \dots, \alpha_n(t)]$$

$$\alpha_i(t) = \text{prob}[O(1), O(2), \dots, O(t) / \text{state } i \text{ at } t]$$

$$\beta^T(t) = [\beta_1(t), \dots, \beta_n(t)]$$

$$\beta_i(t) = \text{prob}[O(t+1), O(t+2), \dots, O(F) / \text{state } i \text{ at } t]$$

$$Pr\{O(1), O(2), \dots, O(F)\} = \sum_{i=1}^n \alpha_i(t) \beta_i(t) \quad (11)$$

In matrix form

$$Pr\{O(1), O(2), \dots, O(F)\} = \Pi^T R(1) T R(2) \dots T R(F) \beta(F) \quad (12)$$

$$\alpha^T(t) = \Pi^T R(1) T B(2) \dots T R(t) \quad (13)$$

$$\beta^T(t) = T R(t+1) T R(t+2) \dots T R(F) \beta(F) \quad (14)$$

In the left-to-right model

$$\Pi^T = [1, 0, \dots, 0] \text{ and } \beta^T(F) = [0, 0, \dots, 1] \quad (15)$$

Recognition Stage

In the recognition stage, the goal is to find which Markov model is most likely given the sequence of observations: $O = [O(1), O(2), \dots]^T$. The computation is reduced basically to finding $\text{prob}[O/M]$ for each model. For this only the set of matrix multiplies in equation (12) need to be carried out.

Throughout this probability calculation, the matrix T remains constant. Also, the matrices $R(t)$ are the same for many consecutive values of t . Consider now equation (12) decomposed as follows:

$$Pr\{O\} = \Pi [R(1) T R(2) \dots R(i) T] \quad (16)$$

$$[R(i+1) T R(i+2) T \dots R(j) T]$$

$$[\dots][\dots][\dots T R(F) \beta(F)]$$

where the sets of matrices within the square brackets correspond to times over which the observations remain constant. The products can be evaluated quite efficiently. Consider the partial product

$$[R(i+1) T R(i+2) T \dots R(j) T] \quad (17)$$

which is equal to

$$[R(j) T]^{j-i} \quad (18)$$

The matrix T is upper triangular and $R(j)$ is diagonal. Therefore $R(j)T$ is upper triangular. If the diagonal elements of $R(j)T$ are distinct, then it can be diagonalized such that:

$R(j)T = P D P^{-1}$ where D is diagonal with its elements the same as the diagonal elements of $R(j)T$. Therefore:

$$[R(j)T]^{j-i} = P D^{j-i} P^{-1} \quad (19)$$

Stated in terms of the partial forward probabilities:

If $O(t+1) = O(t+2) = \dots = O(t+m)$, then

$$\alpha(t+m) = \alpha(t) [T R(m)]^m = \alpha(t+m) = \alpha(t) P D^m P^{-1} \quad (20)$$

Since the matrix P is comprised of orthonormal eigenvectors, its inverse is merely its transpose. Also, since the eigenvalues are the diagonal entries in the upper triangular matrix, the eigenvectors are obtainable by an efficient recursion. Throughout this procedure, as in other hidden Markov model based systems, scaling must be done to ensure no underflow.

Training Stage

For training, we use a variant of the Baum-Welch reestimation procedure. At each iteration, T_{ij} and $b_j(k)$ are estimated based on the previous estimates and the observations. In summary

$$T_{ij} = \frac{\gamma_{ij}}{\gamma_i} \quad (21)$$

$$\hat{b}_j(k) = \frac{\sum_{t \in O(t)=k} \alpha_j(t) \beta_j(t)}{\sum_{t=1}^F \alpha_j(t) \beta_j(t)} \quad (22)$$

$$\text{where } \gamma_{ij} = \frac{1}{P} \sum_{t=1}^{F-1} \alpha_j(t) T_{ij} \hat{b}_j(O(t+1)) \beta_j(t+1) \quad (23)$$

$$\text{and } \gamma_i = \sum_{j=1}^n \gamma_{ij} \quad (24)$$

Here γ_{ij} is the expected number of transitions from state i to j , and γ_i is the number of transitions out of state i .

If these equations were used directly, a large computational burden would exist, since the sequence of observations is so long. However, all equations (21) through (24) denote are sample averages. We therefore do not need to compute the sums over all possible terms, but rather over only a sampling. Denote the modified terms by:

$$\gamma_{ij}^M, T_{ij}^M, \text{ and } \hat{b}_j^M(k).$$

$$\gamma_{ij}^M = \frac{1}{P} \sum_{\tau=1}^{(F-1)/\tau} \alpha_i(\tau) T_{ij} b_j((\tau+1)) \beta_j(\tau+1) \quad (25)$$

$$\gamma_i^M = \sum_{j=1}^F \bar{\gamma}_{ij} \quad \text{and} \quad (26)$$

$$T_{ij}^M = \frac{\gamma_{ij}^M}{\gamma_i^M} \quad (27)$$

Please note that we are now sampling equation (23) every τ units. Similarly,

$$\bar{\gamma}_j^M(k) = \frac{\sum_{t \in (0(\tau+1))=k} \alpha_j(\tau) \beta_j(\tau)}{\sum_{\tau=1}^{F/\tau} \alpha_j(\tau) \beta_j(\tau)} \quad (28)$$

In equation (20) we showed how to compute the values of $\alpha(t+m)$ from $\alpha(t)$ if $0(t+1) = 0(t+2) = \dots = 0(t+M)$. In a similar manner,

$$\beta(t) = [P_{t+1} D_{t+1}^m P_{t+1}^{-1}] \beta(t+m) \quad (29)$$

if $0(t+1) = 0(t+2) = \dots = 0(t+m)$.

Therefore, the recursions for computing the forward and backward partial probabilities can be performed efficiently.

Experiments

To evaluate the system, we used a set of isolated nonsense words differing only in an interior consonant. We chose this task due to the difficulty often encountered in identifying such utterances, and also because we felt our overall procedure was formulated to solve just such problems. The set of words were of the form /*a*-*C*-*i*/ where *C* represents a consonant, including: /b, d, g, p, t, k, r, w, l, j, s, z, f, \theta, z, z, v, y, ts, dz, h, m, n/. These 23 words were spoken twenty times each in two separate sessions (one for training, one for testing) by one male talker. The utterances were filtered and digitized with 12-bit precision at an 8KHz sampling rate.

In parallel with the development system to be tested, a more standard HMM recognition system which accepted as input 6-bit vector quantizer codewords every 10 milliseconds was trained and tested. (This latter system was a highly debugged piece of software developed for other purposes over the last two years.) In both systems, five state left-to-right models were used.

In the standard system, 46 errors were recorded for 90% correct. Twenty-seven of these errors were fricatives being confused with other fricatives. In the new system, 7 errors were recorded for 98.5% correct over the same data. The errors were too few to see a clear trend.

Discussion

We have explored one possible method for approximating a continuous transition hidden Markov model for speech recognition. An important component of this method was to allow a virtually continuous stream of input to be input to the recognizer. The Kalman filter approach is but one of many methods which could be used. We freely admit that we have not at this point explored very deeply into the many variations possible in the Kalman filter model, however. Although our various simplifications led to increased efficiency in the recognition algo-

rithm, the computation time is still a few times larger than with the standard procedure.

On so small a data-base as we have been working, conclusions are perhaps hard to draw. The reduction of the error rate by a factor of 6.5, however, strongly suggests the new method is superior. Although no direct tests were run using continuous density observation probabilities, reported improvements over discrete observation probability models are usually by a much smaller factor.

References

- [1] Lainiotis, D. G., and Park, S. K., "On joint detection estimation, and system identification discrete data case," *Int. Jour. Control*, vol. 17, no. 3, 1973, pp. 609-633.
- [2] Gustafson, D. E., Willsky, A. S., and Wang, J. Y., "ECG/VCG rhythm diagnosis using statistical signal analysis: I. identification of persistent rhythms," *IEEE Trans. Biomed. Eng.*, vol. BME-25, no. 4, July 1978, pp. 344-353.
- [3] Athans, M., Dunn, K. P., Greene, S. C., Lee, W. H., Sandell, N. R., Segall, II, and Willsky, A. S., "The stochastic control of the F-8C aircraft using the multiple model adaptive control (MMAC) method," *Proc. IEEE Decision and Control Conf.*, 10-12, Dec. 1975.

This work was sponsored in part by DoD.

Windowless Analysis of Speech for Automatic Recognition

by

Sungjae Lim

and

Mark A. Clements

Manuscript prepared for the *IEEE Trans. on Acoustics, Speech, and Signal Processing*

Abstract

Traditional hidden Markov model speech recognition is generally based upon a set of parameters which are extracted at discrete intervals. Such an analysis necessitates use of a discrete-transition hidden Markov model in which the underlying states can change only at intervals related to the frame rate of the analysis. The exact locations of the analysis windows can influence the front-end outputs. As a result, inconsistent performance can often be observed in discriminating words which differ only in short duration cues. In the current study, methods are explored which circumvent this framing effect by allowing state transitions to occur at each sample. Efficient methods for implementing this strategy are derived, and testing of a variety of procedures using a set of highly confusable utterances is reported. Significantly superior performance was demonstrated both for quiet and noisy conditions.

1 Introduction

Over the past few years, the method of choice for many speech recognition applications has been on hidden Markov modelling. Steady improvement has been reported in such areas as speaker independence, noise handling, training and response times, as well as general performance. The first HMM based systems modeled speech as a discrete state discrete time Markov process with discrete observations. More recently, models which allow a continuous distribution of observations have been presented. Throughout all these models, however, the assumption remains that sampling the parameterization of the speech (e.g., spectral or LPC based parameters) is only necessary every 10 to 30 milliseconds. When words differ only by a short duration interior consonant, however, the exact placement of the analysis windows can have an impact on performance.

The motivation for the current study came from our observations that although general performance of a recognizer may not depend highly on the exact placement of frames, the detailed error patterns often would. The methods explored are attempts at eliminating the apparent framing artifacts by, in essence, extracting a set of parameters for every sample of the digital speech. The recognition algorithm can then be considered a close approximation to a continuous transition hidden Markov model. This approach would not be feasible were it not for efficient algorithms we have been formulated for this specific problem.

In this paper, we will first discuss the aspects of hidden Markov models which are conducive to this strategy and discuss the issues involved in training, and recognition. Second we will describe three parameter extraction methods, one of which relies on a novel utilization of Kalman filtering, with others two involving more classical procedures. Third, we will examine experimental results and discuss the conclusion which can be drawn.

2 The Hidden Markov Model

A. Definitions :

Consider a discrete state discrete transition hidden Markov model for each pattern to be recognized. Assume the observations are drawn from a finite alphabet of size M , and a new observation is made for every sample of the digital speech. This would imply some form of vector quantizer continuously outputting a codeword sequence. Although the form and implementation of this process will be described in detail in section 3, for all systems considered, enough memory existed in the analysis to produce long sequences of the same codeword in a segment of an utterance. The importance of this result will become apparent below.

Denote the number of states in a model by n .

- π_i = *probability the model starts in state i ,*
- $\underline{\Pi}^T$ = $[\pi_1, \pi_2, \dots, \pi_n]$
- \mathbf{A} = *transition probability matrix, where :*
- a_{ij} = *probability of transition from state i to state j
in one trial; $i, j = 1, 2, \dots, n$.*
- \mathbf{B} = *observation probability matrix where
 b_{jk} = probability of observing codeword k
given state j .*
- $\mathbf{O}(t)$ = *codeword observed at time t , $1 \leq t \leq F$*
- $\mathbf{R}(t)$ = *observation matrix, consisting of :*

$$\mathbf{R}(t) = \text{diag}[b_1(\mathbf{O}(t)), \dots, b_n(\mathbf{O}(t))]$$

For a given model \mathbf{M} , and observations $\mathbf{O}(1), \mathbf{O}(2), \dots, \mathbf{O}(F)$, we define

- $\underline{\alpha}^T(t)$ = $[\alpha_1(t), \dots, \alpha_n(t)]$
- $\alpha_i(t)$ = *prob $[\mathbf{O}(1), \dots, \mathbf{O}(t); \text{state } i \text{ at } t]$*
- $\underline{\beta}^T(t)$ = $[\beta_1(t), \dots, \beta_n(t)]$
- $\beta_i(t)$ = *prob $[\mathbf{O}(t+1), \dots, \mathbf{O}(F); \text{state } i \text{ at } t]$*

Then the probability that we observe the sequence from the model is

$$Pr[\mathbf{O}(1), \dots, \mathbf{O}(F)] = \sum_{i=1}^n \alpha_i(t) \beta_i(t) \quad (1)$$

We can rewrite $\underline{\alpha}(t)$, $\underline{\beta}(t)$, and Eq.(1) in matrix form such that

$$Pr[\mathbf{O}(1), \dots, \mathbf{O}(F)] = \underline{\Pi}^T \mathbf{R}(1) \mathbf{A} \mathbf{R}(2) \mathbf{A} \dots \mathbf{A} \mathbf{R}(F) \underline{\beta}(F) \quad (2)$$

$$\underline{\alpha}^T(t) = \underline{\Pi}^T \mathbf{R}(1) \mathbf{A} \mathbf{R}(2) \dots \mathbf{A} \mathbf{R}(t) \quad (3)$$

$$\underline{\beta}^T(t) = \mathbf{A} \mathbf{R}(t+1) \mathbf{A} \mathbf{R}(t+2) \dots \mathbf{A} \mathbf{R}(F) \underline{\beta}(f) \quad (4)$$

If the model is constrained to the left-to-right, \mathbf{A} will be upper triangular. If the model demands the system to start in state 1 and end in state n , then

$$\begin{aligned} \underline{\Pi}^T &= [1, 0, \dots, 0] \\ \underline{\beta}^T(F) &= [0, \dots, 0, 1] \end{aligned} \quad (5)$$

B. Recognition :

For a given model, one needs to compute the probability of the observations. This can be accomplished, of course, through evaluation of Eq.(2). In our system, F is normally such a large number that direct evaluation of Eq.(2) would require tremendous amount of computation. In order to reduce this computational burden, we make use of the fact that usually a long run of the same codewords are observed, which makes Eq.(2) several long runs of the same matrix multiplications, and the constraint that the model be left-to-right which makes \mathbf{A} upper-triangular. Let's assume that the codewords at time $t+1$ through $t+m$ are same. Then the partial product of Eq.(2) for the period of time,

$$[\mathbf{A} \mathbf{R}(t+1) \mathbf{A} \mathbf{R}(t+2) \dots \mathbf{A} \mathbf{R}(t+m)],$$

is equal to

$$[\mathbf{A} \mathbf{R}(t+m)]^m$$

Since the matrix \mathbf{A} is upper-triangular and $\mathbf{R}(t+m)$ is diagonal, the product, $[\mathbf{A} \mathbf{R}(t+m)]^m$ is an upper-triangular matrix. The upper-triangular matrix has a nice property that it can be diagonalized if the diagonal elements are distinct. In our case, if we assume that the diagonal elements of $\mathbf{A} \mathbf{R}(t+m)$ are distinct, it can be diagonalized in such a form that

$$\mathbf{A} \mathbf{R}(t+m) = \mathbf{P} \mathbf{D} \mathbf{P}^{-1} \quad (6)$$

where \mathbf{D} is diagonal with its elements same as the diagonal elements of $\mathbf{AR}(t + m)$, \mathbf{P} is a upper-triangular matrix with its diagonal elements equal to 1. Therefore,

$$[\mathbf{AR}(t + m)]^m = \mathbf{PD}^m\mathbf{P}^{-1} \quad (7)$$

And $\underline{\alpha}(t + m)$ can be computed directly from $\underline{\alpha}(t)$ without computing intermediate $\underline{\alpha}'$'s at $t + 1, t + 2, \dots$, and $t + m - 1$, that is,

$$\begin{aligned} \underline{\alpha}(t + m) &= \underline{\alpha}(t)[\mathbf{AR}(t + m)]^m \\ &= \underline{\alpha}(t)\mathbf{PD}^m\mathbf{P}^{-1} \end{aligned} \quad (8)$$

It seems that obtaining the matrices, \mathbf{P} and \mathbf{P}^{-1} , require time-consuming computation, especially when the dimension of the matrix is large. This, however, is not so in our case. In fact, there exist very efficient ways using the property that $[\mathbf{AR}(t + m)]$ is upper-triangular. The efficient methods to compute \mathbf{P} and \mathbf{P}^{-1} are shown in Appendix A.

C. Training Algorithms :

In the previous section, we have shown an efficient way of computing $\underline{\alpha}'$'s without computing the intermediate ones when a long run of the same codewords are observed. $\underline{\beta}'$'s can also be computed in the same way. In this section, two different training methods are introduced in which we make use of the same method to efficiently carry out the reestimation. The first one, denoted as "Algorithm 1", is strictly based on the Baum-Welch reestimation algorithm, while the second one, denoted as "Algorithm 2", is slightly varied version and yet performs better.

1). Algorithm 1 :

The Baum-Welch reestimation algorithm states that the estimates of a_{ij} and $b_j(v)$, denoted as \hat{a}_{ij} and $\hat{b}_j(v)$ respectively, are updated at each iteration based on the previous estimates as follows :

$$\hat{a}_{ij} = \frac{\gamma_{ij}}{\gamma_i} \quad (9)$$

$$\hat{b}_j(v) = \frac{\sum_{t \in O(t)=k} \alpha_j(t) \beta_j(t)}{\sum_{t=1}^F \alpha_j(t) \beta_j(t)} \quad (10)$$

where

$$\gamma_{ij} = \frac{1}{p} \sum_{t=1}^{F-1} \alpha_j(t) \hat{a}_{ij} \hat{b}_j(O(t+1)) \beta_j(t+1) \quad (11)$$

$$\gamma_i = \sum_{j=1}^n \gamma_{ij} \quad (12)$$

Let's consider the computation of γ_{ij} . If $O(k+1) = O(k+2) = \dots = O(k+m)$, then $b_j(O(k+1)) = b_j(O(k+2)) = \dots = b_j(O(k+m))$. Thus the partial summation of Eq.(11) for $k \leq t \leq k+m-1$, denoted as $\gamma_{ij}(k, k+m-1)$, can be written as

$$\gamma_{ij} = \frac{1}{p} [a_{ij} b_j(O(k+1))] \sum_{t=k}^{k+m-1} \alpha_i(t) \beta_j(t+1) \quad (13)$$

Computation of Eq.(13) in a straight forward way requires $\alpha_i(t)$ and $\beta_j(t+1)$ to be computed at $t = k, k+1, \dots, k+m-1$. With a different manipulation, which will be shown in the following, this can be avoided and a lot of computation can also be saved, especially when m is large. First let's express $\underline{\alpha}(t)$ and $\underline{\beta}(t+1)$ for $k \leq t \leq k+m-1$ in terms of $\underline{\alpha}(k)$ and $\underline{\beta}(k+m)$ as follows ;

$$\underline{\alpha}^T(t) = \underline{\alpha}^T(k) [\mathbf{AR}(k+1)]^{t-k} \quad (14)$$

$$\underline{\beta}(t+1) = [\mathbf{AR}(k+1)]^{m-t+k-1} \underline{\beta}(k+m) \quad (15)$$

Then

$$\begin{aligned} \sum_{t=k}^{k+m-1} \alpha_i(t) \beta_j(t+1) &= \sum_{t=k}^{k+m-1} [\underline{\alpha}(t) \underline{\beta}^T(t+1)]_{ij} \\ &= \sum_{t=k}^{k+m-1} [((\mathbf{AR})^{t-k})^T \underline{\alpha}(k) \underline{\beta}^T(k+m) ((\mathbf{AR})^{m-t+k-1})^T]_{ij} \end{aligned} \quad (16)$$

where $[*]_{ij}$ denotes i-j component of matrix $[*]$ and $\mathbf{R} = \mathbf{R}(k+1)$ for simplicity. As shown in the previous section, \mathbf{AR} can be decomposed such that $\mathbf{AR} = \mathbf{PDP}^{-1}$. Then Eq.(16) can be rewritten as follows;

$$\begin{aligned} &\sum_{t=k}^{k+m-1} \alpha_i(t) \beta_j(t+1) \\ &= \sum_{t=k}^{k+m-1} [\mathbf{P}^{-T} \mathbf{D}^{t-k} \mathbf{P}^T \underline{\alpha}(k) \underline{\beta}^T(k+m) \mathbf{P}^{-T} \mathbf{D}^{m-t+k-1} \mathbf{P}^T]_{ij} \\ &= [\mathbf{P}^{-T} (\sum_{t=k}^{k+m-1} \mathbf{D}^{t-k} \mathbf{P}^T \underline{\alpha}(k) \underline{\beta}^T(k+m) \mathbf{P}^{-T} \mathbf{D}^{m-t+k-1}) \mathbf{P}^T]_{ij} \end{aligned} \quad (17)$$

If we let

$$\hat{\underline{\alpha}}(k) = \mathbf{P}^T \underline{\alpha}(k) \quad (18)$$

$$\hat{\underline{\beta}}^T(k+m) = \underline{\beta}^T(k+m)\mathbf{P}^{-T}, \quad (19)$$

then Eq.(17) can be written more neatly such that

$$\sum_{t=k}^{k+m-1} \alpha_i(t)\beta_j(t+1) = [\mathbf{P}^{-T}\mathbf{M}\mathbf{P}^T]_{ij}, \quad (20)$$

where

$$\begin{aligned} \mathbf{M} &= \sum_{t=k}^{k+m-1} \mathbf{D}^{t-k} \mathbf{P}^T \underline{\alpha}(k) \underline{\beta}^T(k+m) \mathbf{P}^{-T} \mathbf{D}^{m-t+k-1} \\ &= \sum_{t=k}^{k+m-1} \mathbf{D}^{t-k} \hat{\underline{\alpha}}(k) \hat{\underline{\beta}}^T(k+m) \mathbf{D}^{m-t+k-1} \end{aligned} \quad (21)$$

Now let's consider the computation of \mathbf{M} . The $i - j^{th}$ component of \mathbf{M} , M_{ij} , can be expressed as

$$\begin{aligned} M_{ij} &= \sum_{t=k}^{k+m-1} d_i^{t-k} \hat{\alpha}_i(k) \hat{\beta}_j(k+m) d_j^{m-t+k-1} \\ &= (\hat{\alpha}_i(k) \hat{\beta}_j(k+m)) \sum_{t=k}^{m-t+k-1} d_i^{t-k} d_j^{m-t+k-1} \end{aligned} \quad (22)$$

Since it was assumed that $d_i \neq d_j$ if $i \neq j$, the summation can be reduced such that

$$\sum_{t=k}^{k+m-1} d_i^{t-k} d_j^{m-t+k-1} = \begin{cases} \frac{d_j^m - d_i^m}{d_j - d_i} & \text{for } i \neq j \\ m(d_i)^{m-1} & \text{for } i = j \end{cases} \quad (23)$$

Thus

$$M_{ij} = \begin{cases} \frac{d_j^m - d_i^m}{d_j - d_i} \hat{\alpha}_i(k) \hat{\beta}_j(k+m) & \text{for } i \neq j \\ m d_i^{m-1} \hat{\alpha}_i(k) \hat{\beta}_j(k+m) & \text{for } i = j \end{cases} \quad (24)$$

In summary,

$$\gamma_{ij}(k, k+m-1) = \frac{1}{p} [\mathbf{P}^{-T} \mathbf{M} \mathbf{P}^T]_{ij} (a_{ij} b_j(O(k+1))) \quad (25)$$

It is worth to be noted that only the upper triangular portions of \mathbf{M} are necessary to be computed, since we only need $\gamma_{ij}(k, k+m-1)$, for $i \leq j$ and the matrices, \mathbf{P}^{-T} and \mathbf{P}^T , are lower triangular.

Secondly, let's consider the numerator of Eq.(10) for the reestimation of $b_j(v)$. Under the same assumption that $O(k+1) = O(k+2) = \dots = O(k+$

$m) = v$, the partial summation of the numerator, $\sum_{t \in O(t)=v} \alpha_j(t) \beta_j(t)$, for $k + 1 \leq t \leq k + m$ can be expressed in terms of $\underline{\alpha}(k)$ and $\underline{\beta}(k + m)$,

$$\begin{aligned} \sum_{t=k+1}^{k+m} \alpha_j(t) \beta_j(t) &= \sum_{t=k+1}^{k+m} [\underline{\alpha}(t) \underline{\beta}^T(t)]_{jj} \\ &= [\mathbf{P}^{-T} \sum_{t=k+1}^{k+m} (\mathbf{D}^{t-k} \mathbf{P}^T \underline{\alpha}(k) \underline{\beta}^T(k+m) \mathbf{P}^{-T} \mathbf{D}^{k+m-t}) \mathbf{P}^T]_{jj} \end{aligned} \quad (26)$$

Eq.(26) is very similar to Eq.(17), and can be evaluated similarly. In fact, if we denote the term in the summation of Eq.(26) as $\hat{\mathbf{M}}$, i.e.,

$$\hat{\mathbf{M}} = \sum_{t=k+1}^{k+m} \mathbf{D}^{t-k} \mathbf{P}^T \underline{\alpha}(k) \underline{\beta}^T(k+m) \mathbf{P}^{-T} \mathbf{D}^{k+m-t} \quad (27)$$

It can be observed that $\hat{\mathbf{M}}$ is the product of \mathbf{D} and \mathbf{M} , i.e.,

$$\hat{\mathbf{M}} = \mathbf{D} \mathbf{M} \quad (28)$$

Hence, once \mathbf{M} is obtained to compute $\gamma_{ij}(k, k + m - 1)$, Eq.(22) can be computed with only a few more computation as follows;

$$\sum_{t=k+1}^{k+m} \alpha_j(t) \beta_j(t) = [\mathbf{P}^{-T} \mathbf{D} \mathbf{M} \mathbf{P}^T]_{jj} \quad (29)$$

As mentioned earlier, in the partial summations involved for the restimations of a_{ij} and $b_j(v)$, $\underline{\alpha}'$'s and $\underline{\beta}'$'s are not required to be computed at every time unit. For example, if we consider the assumption given above that $O(t+1) = O(t+2) = \dots = O(t+m)$, only $\underline{\alpha}(k)$ and $\underline{\beta}(k+m)$ are required in the partial summations, that is, all the intermediate $\underline{\alpha}'$'s and $\underline{\beta}'$'s do not have to be computed, which contributes to the great saving of computation.

2). Algorithm 2 :

The algorithm presented here can be considered as the sampling version of Baum-Welch restimation algorithm. Unlike the Baum-Welch algorithm, which is formulated by Eq.(9) and Eq.(10), in the new algorithm only the samples of γ_{ij} are used. Eq.(11) can be rewritten as follows;

$$\gamma_{ij} = \sum_{t=1}^{F-1} \gamma_{ij}(t) \quad (30)$$

where

$$\gamma_{ij}(t) = \frac{1}{p} \alpha_i(t) \bar{a}_{ij} \bar{b}_j(O(t+1)) \beta_j(t+1) \quad (31)$$

The reestimation equations (9) and (10) can also be written in terms of $\gamma_{ij}(t)$.

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{F-1} \gamma_{ij}(t)}{\sum_{j=1}^n (\sum_{t=1}^{F-1} \gamma_{ij}(t))} \quad (32)$$

$$\bar{b}_j(v) = \frac{\sum_{t \in O(t)=v} (\sum_{i=1}^n \gamma_{ij}(t-1))}{\sum_{i=1}^F (\sum_{i=1}^n \gamma_{ij}(t-1))} \quad (33)$$

In the new algorithm, we sample $\gamma_{ij}(t)$ at every k^{th} time unit, and assume that it stays same during the sampling interval. In other words, if $\gamma_{ij}(t)$ is sampled at $t = 1, k+1, 2k+1, \dots$, then we assume that

$$\begin{aligned} \gamma_{ij}(1) &= \gamma_{ij}(2) = \dots = \gamma_{ij}(k) \\ \gamma_{ij}(k+1) &= \gamma_{ij}(k+2) = \dots = \gamma_{ij}(2k) \\ \gamma_{ij}(2k+1) &= \gamma_{ij}(2k+2) = \dots = \gamma_{ij}(3k) \\ &\vdots \end{aligned} \quad (34)$$

Under this assumption and the assumption that $F = mk$ for some integer m , Eq.(32) becomes as follows,

$$\bar{a}_{ij} = \frac{\sum_{\tau=0}^{m-1} \gamma_{ij}(\tau k + 1)}{\sum_{j=1}^n \sum_{\tau=0}^{m-1} \gamma_{ij}(\tau k + 1)} \quad (35)$$

which can be seen as the sampled version. This algorithm is not proven mathematically to converge, but it has shown experimentally that it not only converges but also gives better results than the conventional Baum-Welch algorithm. It seems that this algorithm has a smoothing property which enables the algorithm to find a better local maximum point.

3 Front-End Analysis

The approach we are adopting is based on a linear model of speech which is time-invariant over short intervals. This is the traditional model often used in speech recognition and coding applications. However, we allow for natural smooth changes occurring in the system as well as additive uncorrelated noise. Our linear model may also have explicit modeling of time-varying system parameters. Since many phonemes are characterized by a particular evolution in time rather than by steady-state or target spectra, this model is more

powerful than more traditional ones. In particular our model is :

$$\begin{cases} \mathbf{X}(k) &= \Phi(k)\mathbf{X}(k-1) + \Gamma(k)w(k) \\ s(k) &= \mathbf{H}^T\mathbf{X}(k) + v(k) \end{cases} \quad (36)$$

where the vector $\mathbf{X}(k) = [x(k)x(k-1)\cdots x(k-p+1)]^T$, $x(k)$ is the speech without noise, $w(k)$ the noise input and $\Gamma(k)$ its gain, $\mathbf{H}^T = [1, 0, 0, \dots, 0]$, $v(k)$ the additive noise, and $\Phi(k)$ characterizes the time-varying vocal-tract filter.

Systems similar to this have been used to model many varied signals arising in sonar, heart monitoring, aircraft control, etc.. In the linear prediction synthesis model $\Phi(k)$ remains constant over 10 to 30 millisecond intervals, and $v(k)$ is zero. In the LPC analysis model, $v(k)$ is generally assumed to be zero so that $\Phi(k)$ can be estimated every 10 to 30 milliseconds. Recursive linear least square estimation based on our model falls within the general area of Kalman filtering, which allows one to efficiently compute the least squares estimate of $\mathbf{X}(k)$ from the least squares estimate of $\mathbf{X}(k-1)$ and $s(k)$. The property we wish to exploit is that if we have modeled the system correctly, the prediction error, $\epsilon(k)$, would be white, and it should have a predictable ratio of its power to the unfiltered signal's power. If there are L possible models from which the observed signals was generated, this idea can be used for computing the relative likelihood of each model given the observed signal. In the following our front-end process is explained in detail on the Kalman filtering process followed by decision making process.

3.1 Kalman Filtering

In the Kalman filtering process, we have L distinct competing models, each of which has the form,

$$\begin{cases} \mathbf{X}(k) &= \Phi\mathbf{X}(k-1) + \Gamma(k)w(k) \\ s(k) &= \mathbf{H}\mathbf{X}(k) + v(k) \end{cases} \quad (37)$$

where

$$\Phi = \begin{bmatrix} a(1) & a(2) & \cdots & a(p) \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

$$\mathbf{H} = [100 \cdots 0]$$

$$E v(k) = 0 \quad E v(k) v(l) = \sigma_v^2(k) \delta_{kl}$$

$$E \mathbf{w}(k) = [0, 0, \dots, 0]^T$$

$$E \mathbf{w}(k) \mathbf{w}(l) = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \delta_{kl}$$

$$\Gamma(k) = \begin{bmatrix} g(k) & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

and $a(1), a(2), \dots, a(p)$ are linear prediction coefficients which characterize the model. This model results in the following time-recursive formula which gives the linear least squares estimate of $\mathbf{X}(k)$ given $s(k-1), s(k-2), \dots, s(0)$.

$$\epsilon(k) = s(k) - \mathbf{H} \hat{\mathbf{X}}(k|k-1) \quad (38)$$

$$\sigma_\epsilon^2(k) = \mathbf{H} \mathbf{P}(k|k-1) \mathbf{H}^T + \sigma_v^2(k) \quad (39)$$

$$\mathbf{M}(k) = \frac{1}{\sigma_\epsilon^2(k)} \mathbf{P}(k|k-1) \mathbf{H}^T \quad (40)$$

$$\hat{\mathbf{X}}(k|k) = \hat{\mathbf{X}}(k|k-1) + \mathbf{M}(k) \epsilon(k) \quad (41)$$

$$\hat{\mathbf{X}}(k+1|k) = \Phi \hat{\mathbf{X}}(k|k) \quad (42)$$

$$\mathbf{P}(k|k) = \mathbf{P}(k|k-1) - \mathbf{M}(k) \mathbf{M}^T(k) \sigma_\epsilon^2(k) \quad (43)$$

$$\mathbf{P}(k+1|k) = \Phi \mathbf{P}(k|k) \Phi^T + \Gamma(k) \Gamma^T(k) \quad (44)$$

where $\epsilon(k)$ is the innovations sequence, $\sigma_\epsilon^2(k)$ the variance of the innovations, $\mathbf{M}(k)$ the Kalman gain, and $\mathbf{P}(k|r)$ the covariance of the estimate error $\mathbf{X}(k) -$

$\hat{\mathbf{X}}(k|r)$. The initial condition is given as follows;

$$\begin{cases} \hat{\mathbf{X}}^T(0|0) = [s(0)s(-1) \cdots s(-p+1)] \\ \mathbf{P}(0|0) = \sigma_v^2(0)\mathbf{I} \end{cases} \quad (45)$$

With the innovations sequence obtained from each model, a likelihood test is performed in a recursive manner. If we denote $\epsilon_i(k)$ the innovation produced by model i at time k and $p_i(k)$ the probability that model i generate $s(k)$, then

$$p_i(k) = \frac{N(\epsilon_i(k), \sigma_{\epsilon_i}^2(k))p_i(k-1)}{\sum_{j=1}^L N(\epsilon_j(k), \sigma_{\epsilon_j}^2(k))p_j(k-1)} \quad (46)$$

where $\sigma_{\epsilon_j}^2(k)$ is the variance of $\epsilon_j(k)$ when model j is correct, and $N(a, b)$ represents the Gaussian density of zero mean with the variance b evaluated at a . We then choose the model with the largest p .

4 Experiments

Several recognition experiments were performed with clean speech, noisy speech of $SNR = 26dB$, and of $SNR = 20dB$. The isolated words used in the experiments are 'break', 'change', 'degree', 'eight', 'eighty', 'enter', 'fifty', 'fix', 'six', 'go'. Each word has 12 utterances, 6 of which were used for the training of HMM's. Each utterance was passed through Kalman-filtering process with 3 different level of white Gaussian noises as stated above, which produced 3 different sets of codewords, one for clean speech, one for the noisy speech of $SNR = 26dB$, and one for the noisy speech of $SNR = 20dB$. In the Kalman-filtering process, the variances of the generating noise and the additive noise were updated at every 80 samples, and the initial conditions were reset accordingly at the same time. The filter order was 14 for each of the 64 different filters.

A. With Clean Speech : 2 errors out of 120 = 1.7 1 error from the set used for training : 'six' recognized as 'fix'

1 error from the set not used for training : 'eight' recognized as 'eighty'

B. With Noisy Speech of $SNR = 26dB$: 8 errors out of 120 = 6.7 0 error

from the set used for training

8 errors from the set not used for training : 'eight' recognized as 'eighty' (4), 'fix' recognized as 'six' (1), 'six' recognized as 'fix' (3).

C. With Noisy Speech of $SNR = 26dB$ and Clean Speech :

i). the recognition of noisy speech : 6 errors out of 120 = 5.0 error from the set used for training

6 errors from the set not used for training : 'eight' recognized as 'eighty' (3), 'fix' recognized as 'six' (1), 'six' recognized as 'fix' (2)

ii). the recognition of clean speech : 7 errors out of 120 = 5.82 errors from the set used for training : 'eight' recognized as 'eighty', and 'six' recognized as 'fix'

5 errors from the set not used for training : 'eight' recognized as 'eighty' (5)

It is interesting to note that the models trained with both clean and noisy speech give higher recognition rate for noisy speech (compare the results of B and C i).) than the ones trained with only noisy speech, while giving lower recognition rate for clean speech (compare the results of A and C ii).) than the ones trained with only clean speech. It may be interpreted as clean speech giving positive information for the training of noisy speech models, and noisy speech giving negative information for the training clean speech models. This behavior has been observed in several occasions. More comprehensive experiments are to be done with larger vocabulary and various SNR's.

Iterative Speech Enhancement With Spectral Constraints

John H. Hansen and Mark A. Clements

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia 30332

Abstract

A new and improved iterative speech enhancement technique based on spectral constraints is presented in this paper. The iterative technique, originally formulated by Lim and Oppenheim, attempts to solve for the maximum likelihood estimate of a speech waveform in additive white noise. The new approach applies inter- and intra-frame spectral constraints to ensure convergence to reasonable values and hence improve speech quality. An extremely efficient technique for applying these constraints is in the use of line spectral pair (LSP) coefficients. The inter-frame constraints ensures more speech-like formant trajectories than those found in the unconstrained approach. Results from speech degraded by additive white Gaussian noise show noticeable quality improvement.

Introduction

The successfulness of an enhancement algorithm rests on the goals and assumptions used in deriving the approach. Depending on the application, a system may be directed at one or more objectives such as improving overall quality, increasing intelligibility, reducing listener fatigue, etc. Three assumptions normally made include: i) that the noise distortion be additive, ii) that only the degraded speech signal is available, and iii) that the noise and speech signals are uncorrelated. In general, constraints placed on the speech model improve the potential for separating speech from background noise. However, such systems are also more sensitive to "deviations" from these constraints. The degradation considered is additive white Gaussian noise. The basis of the technique is an iterative enhancement approach based on noncausal Wiener filtering originally formulated by Lim and Oppenheim [1]. This approach attempts to solve for the maximum likelihood estimate of a speech waveform in additive white noise using the constraint that the signal is an all-pole process. Crucial to the success of this approach is the accuracy of the estimates of the all-pole speech parameters at each iteration. One advantage of the Wiener filtering approach is that no "musical tone" artifacts are present after processing as can be observed in spectral subtraction techniques. In addition, under certain conditions, it can be shown that it is the optimal solution in the mean-squared sense for a white noise distortion. Although successful in a mathematical sense, this technique has received little application due to several factors. First, it is an iterative scheme with sizable computational requirements as opposed to a direct form such as spectral subtraction. Second, although the original sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, heuristic convergence criteria had to be employed. After an extensive investigation [2], this approach was found to produce significant levels of enhancement for white Gaussian noise in 3-4 iterations. The technique was generalized to allow for colored aircraft noise. Various spectral estimation techniques were employed for securing estimates of the colored background noise and although the noise was not stationary, estimates were performed prior to application of the algorithm.

With these assumptions, good enhancement took place in 2-3 iterations. It is assumed that in a real-time environment however, noise spectral estimates could be gathered and updated during silent intervals. An important observation which could be made from this previous work was that as additional iterations were performed, individual formants of the speech decreased in bandwidth (see fig.1), resulting in unnatural sounding speech. Frame-to-frame pole jitter was also observed which contributed to unnatural sounding results. Also, the original technique employs no explicit frame-to-frame constraints. Since the original algorithm already constrains the speech to be the response from an all-pole system, applying further constraints on the pole movements may improve the algorithms performance. One set of constraints were applied directly to the LPC poles. These results were quite encouraging, yet computationally intensive. A new approach for implementing the spectral constraints was formed by employing the line spectral pair (LSP) transformation as a method for representing the vocal tract spectrum. This method of specification allowed constraints to be efficiently applied to the speech model pole movements across time (inter-frame) so that formants lay on smooth tracks. In addition, constraints could also be easily applied across iterations (intra-frame) on a frame-by-frame basis.

Iterative Speech Enhancement

Enhancement based on the estimation of all-pole speech parameters in additive white Gaussian noise was investigated by Lim and Oppenheim [1], and later for a colored noise degradation by Hansen and Clements [2]. It was shown that the estimation procedures which result in linear equations without background noise, become nonlinear when noise is introduced. However by allowing a suboptimal procedure, an iterative algorithm results which possesses the property that the estimation procedure is linear at each iteration.

Consider the statistical parameter estimation of speech in the presence of noise. Over a short-time basis, the speech signal can be represented as the following difference equation:

$$s(n) = \mathbf{a}^T \mathbf{s}(n-1, n-p) + g w(n) \quad (1)$$

where $\mathbf{a}^T = [a_1, a_2, \dots, a_p]$ represents the all-pole predictor coefficients. Substituting the degraded speech into the speech model gives the following equation for the observation vector:

$$\mathbf{y}_0 = \mathbf{y}(N-1, 0) = \mathbf{s}(N-1, 0) + \mathbf{d}(N-1, 0) \quad (2)$$
$$\mathbf{y}_0 = \mathbf{a}^T \mathbf{y}(n-1, n-p) + g w(n) + \mathbf{d}(n) - \mathbf{a}^T \mathbf{d}(n-1, n-p)$$

where $\mathbf{s}(N-1, 0)$ are N samples of original speech, and $\mathbf{d}(N-1, 0)$ represents the additive background noise. The $2p + 1$ unknowns include the predictor coefficients \mathbf{a} , initial conditions for the predictor given by $\mathbf{S}_1 = \mathbf{s}(-1, -p)$, and the gain factor g for the input excitation. Consider the case where all unknown parameters are random with a priori Gaussian probability density functions. The basic procedure used is a maximum a priori (MAP) estimator, which maximizes the probability density function of

the parameters given the observations. Therefore, a, g, S_1 are chosen to maximize the probability density function $p(a, g, S_1 | Y_0)$. The procedure requires that a be chosen to maximize $p(a | Y_0)$, noting that the estimate is conditioned on the noisy observations Y_0 . Using Bayes' rule, $p(a | Y_0)$ can be written as a product of terms involving $p(Y_0 | a, g, S_1)$. When the Gaussian density function $p(Y_0 | a, g, S_1)$ is expanded, it can be shown that the mean and variance are functions of the predictor coefficients a . Therefore the resulting equations for maximizing $p(a | Y_0)$ are nonlinear, involving partial derivatives with respect to a . Lim and Oppenheim considered a suboptimal solution employing a two step approach based on MAP estimation of S_0 given Y_0 , followed by MAP estimation of a given \hat{S}_0 , where \hat{S}_0 is the result of the first estimation. Observations indicate that this algorithm converges to a local maximum of the joint density $p(a, S_0 | Y_0, g, S_1)$. In particular, if the probability density function is unimodal, and the initial estimate for a is such that the local maximum equals the global maximum, then the procedure is equivalent to the joint MAP estimate of a and S_0 . After some simplification, the MAP estimation of S_0 , based on maximizing the probability density function $p(S_0 | a, Y_0)$ which is jointly Gaussian in Y_0 , is equivalent to a minimum mean squared error (MMSE) estimate of S_0 . Therefore as the observation window increases in length, the procedure for obtaining a MMSE estimate of $s(n)$ approaches a noncausal Wiener filter. With this, the implementation of the algorithm is presented in Figure 2. This approach can also be extended to the colored noise case as shown. As indicated, the background noise spectral density must be estimated during non-speech activity.

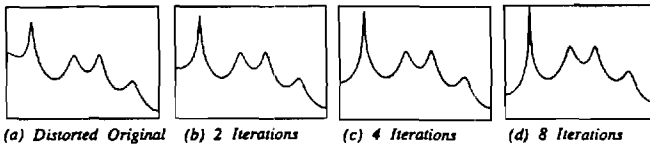


Figure 1: Variation in vocal tract response across iterations.

As indicated, the sequential MAP estimation technique increases the joint likelihood of the speech waveform and all-pole parameters, yet a heuristic convergence criterion had to be employed. Also, as additional iterations were performed, individual formants of the speech decrease in bandwidth as indicated in figure 1. Frame-to-frame pole jitter was also observed. Both effects contributed to unnatural sounding speech. The goal, therefore is to impose constraints on the pole movements across time (inter-frame) and iterations (intra-frame). An initial approach was to limit the poles from moving too close to the unit circle by performing an off-axis spectral evaluation where the z -transform is evaluated on a circle further away from the poles of the spectral model. Other approaches considered included applying constraints directly to the pole radii and/or angular displacements in the LPC model. Performance of such inter and intra-frame constraints lead to encouraging results, but at the expense of a p th order root-solve and a pole ordering step per frame for each iteration. Since root solving is not always numerically accurate and ordering can be inconsistent across frames, a more robust approach was sought to implement these constraints. Previous success of the line spectral pair (LSP) transformation in speech coding by Crosser [3], led to the use of LSP's for this purpose.

Line Spectral Pair Representation of Spectral Characteristics

The LSP transformation may be viewed as an alternative representation of the LPC spectrum. The LSP coefficients are obtained from the LPC prediction coefficients by combining the forward and backward predictor polynomials as follows:

$$P(z) = A(z) + B(z), \quad Q(z) = A(z) - B(z). \quad (3)$$

The vocal tract transfer function is given by $g/A(z)$, and M is the order of the LPC speech model. The resulting polynomials $P(z)$ and $Q(z)$, are symmetric and antisymmetric, respectively, with a root of $P(z)$ at $z=+1$, and a root of $Q(z)$ at $z=-1$. The remainder of the roots of P and Q all lie on the unit circle. Since the roots occur in conjugate pairs, the original polynomial can be represented by M real numbers. The angles of the roots, $\{\omega_i, i=1,2,\dots,M\}$, are called the *line spectrum pairs*.

The LSP's possess several important properties which make them attractive for use in applying spectral constraints. One important characteristic is that if the vocal tract polynomial $A(z)$ has all its roots inside the unit circle (i.e., a stable filter), then the roots of P and Q will be interleaved around the unit circle [3]. If two adjacent LSP frequencies are identical, it indicates that a root of $A(z)$ lies on the unit circle.

In addition to their attractive representation of the LPC spectrum, the LSP coefficients offer the possibility of a more direct representation of perceptually important information. Specifically, there is a firm statistical relationship between the locations and bandwidths of the speech formants and the locations of the roots of P and Q respectively. Since roots of the P polynomial correspond approximately to locations of formant center frequencies (when a formant is present), the P polynomials' LSP coefficients are termed *position coefficients*. It can be shown that the closer two LSP coefficients are together, the narrower the bandwidth of the corresponding pole of the vocal tract filter. Therefore, formants are indicated when two LSP coefficients are close together. When LSP coefficients are far apart, they indicate poles which contribute only to the overall spectral shape. Because of their relationship to the presence or absence of a formant by their nearness to a position coefficient, the coefficients of Q are termed *difference coefficients*. Given the LSP coefficients, the position coefficients are simply the odd index LSP coefficients, $\{\omega_{2i-1}, i=1,2,\dots,M/2\}$. The difference coefficients are given as follows:

$$d_i = \text{MIN}_{j=-1,1} (|\omega_{2i+j} - \omega_{2i}|), \quad i = 1,2,\dots,M/2 \quad (4)$$

where the sign of d_i is positive if ω_{2i} is closer to ω_{2i+j} , and otherwise is negative. With this interpretation, a new enhancement technique based on Wiener filtering is now possible by imposing constraints on the LSP coefficients.

Step 1: Estimate a_1 from $S_{0,1}$.

Use either: i. first P values as the initial condition vector or: ii. always assume $S_1 = 0$.

Step 2: i. Using a_1 , estimate the speech spectrum:

$$P_s(\omega) = \frac{g^2}{|1 - \sum_{k=1}^p a_k e^{j\omega k}|^2}$$

ii. Calculate gain term using Parseval's theorem.

iii. Estimate either the degrading

a.) white noise variance σ_d^2 , or b.) colored noise spectrum $P_D(\omega)$ from a period of silence closest to the utterance.

iv. Construct the noncausal Wiener filter;

$$\text{a.) } H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + \sigma_d^2} \quad \text{b.) } H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_D(\omega)}$$

v. Filter the estimated speech \hat{s}_1 to produce \hat{s}_{1+1} .

vi. Repeat until some specified error criterion is satisfied, $\Delta e < \text{THRESHOLD}$.

Figure 2: Enhancement Algorithm based on All-pole modeling/Wiener filtering. a) a AWGN distortion b) a non-white distortion

Enhancement with Spectral Constraints

Consider the statistical parameter estimation of speech in the presence of noise, where all unknown parameters are random with a priori Gaussian probability density functions. It can be shown that MAP estimation of a , g , and S_1 given the noisy observations Y_0 , results in a set of nonlinear equations. Therefore, instead of joint estimation of a and S_0 , a suboptimal solution is formulated employing a two step approach based on

MAP estimation of S_0 given Y_0 , followed by MAP estimation of \hat{a} given \hat{S}_0 , where \hat{S}_0 is the result of the first estimation. Since speech can be considered short-time stationary, frame-to-frame spectral constraints may aid in enhancement. The new approach imposes such constraints on the vocal tract spectrum between MAP estimation steps. The procedure for obtaining the MAP estimate of \hat{a} from $\text{MAX } p(\hat{a}|\hat{S}_0;g,S_1)$ remains the same. The next step is to apply spectral constraints to \hat{a}_1 which will ensure that; i) the all-pole speech model is stable, ii) it possess speech-like characteristics (i.e., poles are not too close to the unit circle causing narrow bandwidths), and iii) the vocal tract characteristics do not vary wildly from frame-to-frame when speech is present. Due to this constrained approach, an improved estimate \hat{a}_1 results. Given this new estimate, the second MAP estimation of S_0 given \hat{a}_1 can be carried out by maximizing $p(S_0|\hat{a}_1,Y_0;g,S_1)$. Since $p(S_0|\hat{a}_1,Y_0;g,S_1)$ is still jointly Gaussian in Y_0 , the resulting MAP estimate is equivalent to a MMSE estimate of S_0 . Again, in the limiting case, the procedure for obtaining the MMSE estimate of $s(n)$ approaches a noncausal Wiener filter. Once this new estimate of $\hat{S}_{0,1}$ is formed, the iterative procedure continues by re-estimating \hat{a}_1 , applying constraints to \hat{a}_1 , and then forming the noncausal filter using \hat{a}_1 to re-estimate $\hat{S}_{0,1}$. This continues until some convergence criterion is satisfied. The procedure for implementing these constraints will now be addressed.

Two classes of spectral constraints are considered; inter-frame (across time), and intra-frame (across iterations). Two approaches are considered: a fixed frame rate, and a variable frame rate approach. In the first of these, the LPC predictor coefficients, \hat{a} , are first converted to LSP position and difference coefficients. Next, each frame's energy is observed, and if it is above some threshold, it is classified as voiced speech; if it is below, then it is either noise or unvoiced speech. A local running count L_1 , is kept for the number of consecutive frames which fall below the energy threshold. If L_1 reaches L_{MAX} , then all subsequent frames below the threshold are classified as noise. This allows for further smoothing for long periods of silence. The position coefficients for each frame are smoothed using a weighted triangular window with a variable base of support (1 to 5 frames). If a frame has been classified as noise, maximum smoothing is performed. In addition, the lower formant frequencies are smoothed over a narrower triangle width than for those position coefficients at higher frequencies. This preserves perceptually important speech characteristics found in the lower formants. No smoothing is performed on the difference coefficients since they are more closely related to formant bandwidth than formant location. However, it is possible that a difference coefficient falls within a "forbidden zone," (i.e., the region within d_{MIN} of a position coefficient). When this occurs, the LPC analysis has most likely overestimated the Q of a particular pole. Since this causes unnatural sounding speech, (as in the unconstrained approach), the value of $|d_i|$ is set to d_{MIN} . Finally, the position and difference coefficients are combined to form the constrained LPC predictor coefficients \hat{a}_1 .

The second inter-frame constraint approach considered is a variable frame rate technique which takes advantage of the interpolation properties of the LSP coefficients. The speech signal is first divided into segments, where segments are chosen such that they are long when the speech spectrum is varying slowly and short when the speech spectrum is varying quickly. The LSP coefficients are reconstructed with linear interpolation used to compute the coefficients for intermediate frames.

The segmentation algorithm begins with a step to determine the onset/offset of speech. This is carried out by thresholding the LPC residual energy, which produces relatively long segments. Next, the long segments are subdivided based on the curvature of the position coefficients. This is performed by computing a gain-normalized Itakura-Saito measure of the spectral distance between the frequency response of two adjacent frames. The procedure continues by computing the distortion of

position coefficients for successively longer segments until the distortion exceeds a threshold T_D . At that point, a subsegment boundary is set, with the intermediate position coefficients reconstructed via linear interpolation. During this step, the length of a subsegment is also limited to L_{MAX} to prevent excessively long segments which might contribute to muffled or unnatural sounding speech. The advantage of this approach is that it incorporates more information from adjacent frames when the spectrum indicates similar characteristics. Yet, it also reduces the effects of adjacent frames when the spectrum is significantly different as in the case of a transition from unvoiced passages to noise. This in effect, distorts the position coefficients as little as possible when associated difference coefficients indicate the presence of formants. Difference coefficients for each frame, (or an average set across a segment) are used to compute the predictor coefficients \hat{a}_1 . The difference coefficients are required to be at least d_{MIN} or greater in distance from adjacent position coefficients to ensure that poles from the LPC filter do not move too close to the unit circle.

Inter-frame constraints are applied to a single frame across iterations, and as such require the frames' previous estimates to be available. The motivation for such constraints is that under certain conditions, pole locations for the same frame vary significantly from their previous estimated values. Since the present estimate of \hat{a}_1 affects the next estimate of $\hat{S}_{0,1}$, sections of $\hat{S}_{0,1}$ will also vary significantly across iterations. In addition, previous results based on objective speech quality measures indicated that the unconstrained approach produced minimum objective measures at different iterations for different classes of speech. For example, maximum overall speech quality was observed for additive white Gaussian noise in three iterations. This was also true for vowels and fricatives. However, glides required two iterations, nasals, liquids, and affricates between five and six. It is therefore desirable to be able to affect the convergence rate so that the best objective measure of quality occurs at the same iteration across all classes of speech. Improved quality as measured by objective measures may also result in improved estimation of \hat{a}_1 . By constraining the vocal tract filter to be a function of its previous estimates, it may be possible to accomplish this. Two approaches are considered, one applied to the autocorrelation lags, the other to the position coefficients. The first approach simply weights the present set of autocorrelation lags with the same frame from previous iterations. This technique is very easy to perform, since the autocorrelation lags must be computed in order to estimate the predictor coefficients \hat{a} . The second approach weights position coefficients with those from the same frame but previous iteration. If the corresponding difference coefficient indicates the adjacent position coefficient to represent a formant, this approach has the effect of constraining the formants to lie along smooth tracks across iterations.

Results

Speech degraded by additive white Gaussian noise was processed using various configurations of the new constrained enhancement algorithm. Energy thresholds for inter-frame constraints were obtained from frame energy histograms at each signal-to-noise ratio. Excellent enhancement resulted for a wide range of threshold values. Intra-frame constraints were applied across two to three iterations. Informal listening tests indicated noticeable quality improvement, although no intelligibility testing has been performed. However, there has been extensive work carried out in the area of objective speech quality measures [4]. Good correlation has been shown to exist between subjective quality and objective measures. Therefore, objective measures including: the Itakura-Saito likelihood ratio, log area ratio, and weighted spectral slope measure were used for evaluation. Figure 3 illustrates a comparison of

typical results for the various constraint approaches. Itakura-Saito measure is plotted versus signal-to-noise ratio for a white noise distortion. Plot *a* represents the original distorted speech. Plots *b* through *e* represent combinations of inter-frame constraints (both fixed and variable rate), and intra-frame constraints (applied to position coefficients/autocorrelation lags). All configurations examined showed significant improvement in Itakura-Saito measures. Threshold settings for the variable frame rate inter-frame constraint were somewhat sensitive to varying noise levels. However, the fixed frame approach by itself, and with either autocorrelation or position intra-frame constraints gave impressive results with little sensitivity to varying levels of SNR. In order to determine a limit on the level of enhancement, the original undistorted predictor coefficients a were used in the unconstrained algorithm. In essence, the two step MAP estimation approach is now reduced to a single MAP estimate of S_0 , and therefore represents the theoretical limit for enhancement using Wiener filtering. Plot *f* indicates this limit. Although only Itakura-Saito measures are shown, similar improvement was also observed for log area ratios and weighted spectral slope measures. Figure 4 compares the new approach to existing techniques. Plot *b* shows results from spectral subtraction as formulated by Boll [5]. An evaluation was performed for both half and full-wave rectification, along with one to five frames of magnitude averaging; where these points represent the best results. Plot *c* is from the unconstrained Wiener filtering technique. Plots *d* and *e* are typical values for the inter-frame constraint (fixed frame rate), and inter plus intra-frame constraints (fixed frame and autocorrelation lags). Again *f* indicates the limit for the Wiener filtering approaches.

Sound Type	Itakura-Saito Likelihood Measure			
	Original	Lim-Oppenheim	Hansen-Clements	True LPC
Silence	1.634	1.649	0.842	0.319
Vowel	4.020	3.299	1.651	0.582
Nasal	19.814	17.656	3.968	0.324
Stop	7.261	3.979	1.099	0.435
Fricative	3.739	3.509	1.766	0.649
Glide	1.525	1.442	1.131	0.705
Liquid	9.597	4.545	0.998	0.303
Affricate	3.924	2.702	2.229	0.323
Voiced + Unvoiced	5.838	4.293	1.761	0.519
Total	4.022	3.151	1.364	0.433

SNR=+5dB

Table 1: Comparison of algorithms over sound types for white Gaussian noise.

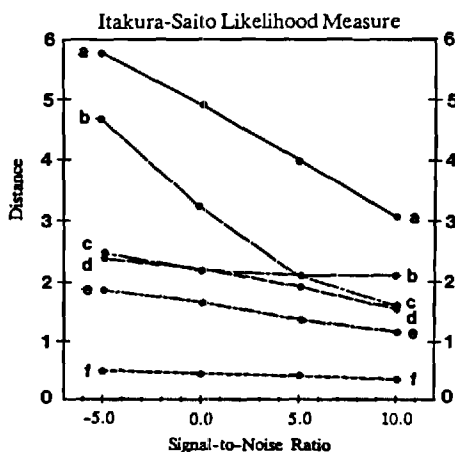


Figure 3: Comparison of constraint algorithms over SNR.

- a.) Original Distorted Speech
- b.) Inter-Frame Constraint: Variable Frame
- c.) Inter-Frame Constraint: Fixed Frame
- d.) Inter & Intra-Frame Constraints: Fixed Frame, Position
- e.) Inter & Intra-Frame Constraints: Fixed Frame, Autocorrelation
- f.) Theoretical limit: using undistorted LPC coefficients, a .

Performance evaluation over sound classes was accomplished by hand partitioning speech into segments. Entire sentences were processed, and objective measures from each class were computed. Table 1 summarizes this comparison between the unconstrained Lim-Oppenheim technique to that of the inter and intra-frame constraint approach. Measures for the theoretical limit using undistorted LPC predictor coefficients a are also indicated. Improvement is indicated for all types of speech. In addition, the constrained approach produced superior objective measures of quality across all speech classes at the same iteration. These results clearly indicate improvement over the unconstrained approach as well as spectral subtraction for additive white Gaussian noise.

Conclusions

The application of spectral constraints to noncausal Wiener filtering results in improved speech enhancement. Informal listening tests along with objective measures such as Itakura-Saito and log-area-ratio's show improvement over the unconstrained technique. By using the Line Spectral Pair transformation, a modest increase in computational requirements results in significant improvement in speech quality. This approach to pole movement constraints is quite robust over direct methods applied to pole radial/angular movements. Finally, this approach may be useful in enhancement for human listeners as well as a preprocessor for speech recognition.

References

- [1] J.S. Lim, A.V. Oppenheim, "All-Pole Modeling of Degraded Speech," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-26, pp. 197-210, June 1978.
- [2] J.H. Hansen, M.A. Clements, "Enhancement of Speech Degraded By Non-White Additive Noise," Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [3] J.R. Crosmer, "Very Low Bit Rate Speech Coding Using the Line Spectrum Pair Transformation of the LPC Coefficients," Ph.D. dissertation, School of Electrical Engineering, Georgia Institute of Technology, Atlanta, June 1985.
- [4] S.R. Quackenbush, "Objective Measures of Speech Quality," Ph.D. dissertation, School of Electrical Engineering, Georgia Institute of Technology, Atlanta, May 1985.
- [5] S.F. Boll, "Suppression of Acoustic Noise in Speech using Spectral Subtraction," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-27, pp. 113-120, April 1978.

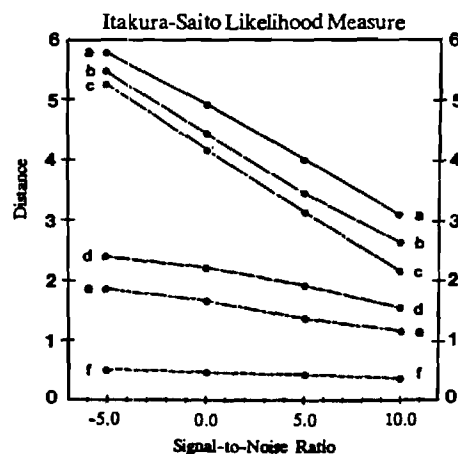


Figure 4: Comparison of enhancement algorithms over SNR.

- a.) Original Distorted Speech
- b.) Boll: Spectral Subtraction, using magnitude averaging
- c.) Lim-Oppenheim: Unconstrained Wiener filtering
- d.) Hansen-Clements: employing Inter-Frame constraints
- e.) Hansen-Clements: employing Inter & Intra-Frame constraints
- f.) Theoretical limit: using undistorted LPC coefficients, a .

Constrained Iterative Speech Enhancement with Application to Automatic Speech Recognition

S12.9

John H. L. Hansen and Mark A. Clements
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, GA 30332

1 Abstract

A set of iterative speech enhancement techniques employing spectral constraints is extended and evaluated in this paper. The original unconstrained technique attempts to solve for the maximum likelihood estimate of a speech waveform in additive noise. The new approaches (presented in ICASSP-87 [3]), apply inter- and intra-frame spectral constraints to ensure optimum speech quality across all classes of speech. Constraints are applied based on the presence of perceptually important speech characteristics found during the enhancement procedure. Previous results show improvement over past techniques for additive white noise distortions. Three points are addressed in the present study. First, a convenient and consistent terminating point for the iterative technique is presented which was previously unavailable. Second, the techniques have been generalized to allow for slowly varying, colored noise. And finally, a comparative evaluation was performed to determine their usefulness as preprocessors for recognition in extremely noisy environments in the vicinity of 0 dB SNR.

2 Introduction

The general problem of automatic speech recognition is one which requires several alternatives to be specified prior to formulation of a solution. The type of speech, restrictions on speakers, vocabulary size, and environment all ultimately affect recognition performance. The specific problem of limited vocabulary, speaker dependent, isolated word recognition has to varying degrees been solved. In the past, approaches such as dynamic time warping or hidden Markov modeling have largely been applied in tranquil environments. Studies have shown that recognition accuracy is severely reduced when speech is uttered in noisy, stressful environments. One alternative is to reformulate previous approaches to the recognition problem assuming a noisy environment. Unfortunately, many systems are LPC based which, from research in speech enhancement and coding are known to deteriorate rapidly in noise. Another alternative, which would be beneficial for recognition as well as speech transmission systems is to develop robust enhancement preprocessors. Such preprocessors would produce speech or recognition features which are less sensitive to background noise so that existing recognition systems may be employed.

The set of speech enhancement algorithms under consideration were previously developed for improving both speech quality and all-pole speech parameter estimation [3,4]. The basis of these algorithms is to form a maximum likelihood estimate of the speech waveform in additive noise with the constraint that the signal be an all-pole process. In section 3, a review of the constrained techniques is presented. A comparative evaluation is presented in sec-

tion 4 which include; additive white Gaussian noise, and slowly varying colored aircraft interior noise. Finally, the enhancement algorithms are evaluated to determine their ability as preprocessors for automatic recognition in extremely noisy environments.

3 Iterative Speech Enhancement

The success of a speech enhancement algorithm is dependent on the objectives made in deriving an approach. Assumptions made in this environment include: i) the noise distortion is additive, ii) only the degraded speech signal is available, and iii) the noise and speech signals are uncorrelated. The basis of the original unconstrained iterative enhancement approach is noncausal Wiener filtering [5]. This approach attempts to solve for the maximum likelihood estimate of a speech waveform in additive white Gaussian noise with the requirement that the signal be the response from an all-pole process. Crucial to the success of this approach is the accuracy of the estimates of the all-pole parameters at each iteration. The algorithm is formulated by considering the case where all unknowns (all-pole speech parameters \vec{a} , noise free speech \vec{S}_0) are random with a priori Gaussian probability density functions. The basic procedure used is a maximum a posteriori (MAP) estimator, which maximizes the probability density function of the unknown parameters given the noisy observations. After some simplification, it can be shown that the resulting equations for the joint MAP estimate of \vec{a} and \vec{S}_0 become nonlinear, involving partial derivatives with respect to \vec{a} . Lim and Oppenheim considered a suboptimal solution employing a sequential two step approach based on MAP estimation of \vec{S}_0 followed by MAP estimation of \vec{a} given $\vec{S}_{0,i}$, where $\vec{S}_{0,1}$ is the result of the first estimation. This sequential estimation procedure is linear at each iteration, and continues until some convergence criterion is satisfied. After further simplifying assumptions, it can be shown that the MAP estimation of \vec{S}_0 is equivalent to a minimum mean squared error (MMSE) estimate. In addition, as the observation window increases, the procedure for obtaining a MMSE estimate approaches a noncausal Wiener filter.

Although successful in a mathematical sense, this technique has received little application due to several factors. First, the scheme is iterative with sizable computational requirements. Second and most important, is that although the original sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, a heuristic convergence criterion had to be employed. This is a serious drawback if the approach is to be used in environments requiring automatic speech enhancement. After an extensive investigation [1], this approach was found to produce significant levels of enhancement for white Gaussian noise in 3-4 iterations. Some interesting anomalies were noted

which helped motivate development of the constrained approaches. First, as additional iterations were performed, individual formants of the speech decreased in bandwidth and shifted in location. Second, frame to frame pole jitter was observed across time. Both effects contributed to unnatural sounding speech. The goal therefore was to formulate a new set of enhancement algorithms which impose constraints on pole locations across time (inter-frame) and iterations (intra-frame). Spectral constraints are applied to the all-pole parameters \hat{a}_i which ensure that; i) the all-pole speech model is stable, ii) it possess speech-like characteristics (e.g., poles are not too close to the unit circle causing narrow bandwidths), and iii) the vocal tract characteristics do not vary wildly from frame to frame when speech is present. Due to the constraints imposed, improved estimates of \hat{a}_{i+1} result. Given this new estimate, the second MAP estimation of \hat{S}_O can be carried out. In order to increase numerical accuracy, reduce computational requirements, and eliminate inconsistencies in pole ordering across frames, the line spectral pair (LSP) transformation was used to implement most of the constraint requirements. Figure 1 illustrates the framework for the constrained enhancement algorithms.

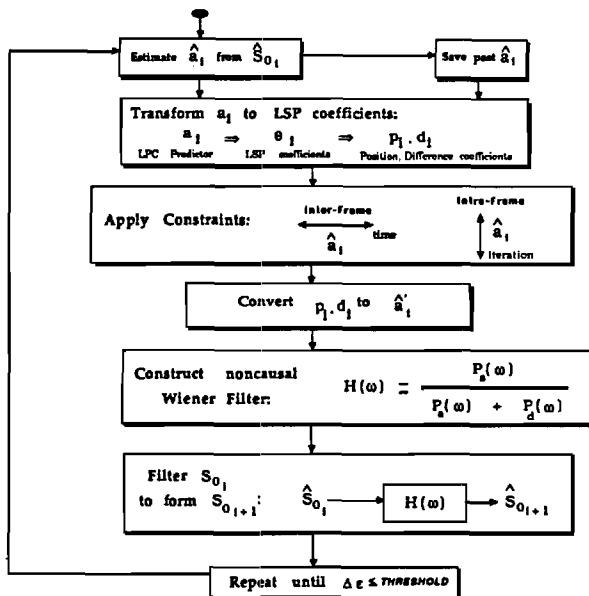


Figure 1: Framework for the constrained iterative enhancement algorithms.

4 Evaluation

Speech degraded by additive noise was processed using various configurations of the constrained algorithms. Enhancement algorithms evaluated include: algorithms incorporating inter-frame constraints applied on a fixed-frame (FF-LSP:T) or variable-frame (VF-LSP:T) basis to the LSP coefficients, algorithms incorporating intra-frame constraints applied to autocorrelation coefficients (Auto:I) or LSP coefficients (LSP:I), along with combinations (FF-LSP:T,Auto:I), (FF-LSP:T,LSP:I), (VF-LSP:T,LSP:I). In the evaluation, global estimates of SNR were employed since the assumption of accurate local estimates is normally unrealistic in actual enhancement environments. Also, energy thresholds for inter-frame constraints were obtained from frame energy histograms at each SNR. In this study, the primary tool for quantitative enhancement

evaluation has been objective quality measures. This is based on extensive work carried out in the formulation of objective speech quality measures [6], and the application of these measures to enhancement [2]. Fair to good correlation has been shown to exist between subjective and objective quality measures.

Evaluation Using Additive White Gaussian Noise

As previously reported, the constrained enhancement algorithms have been shown to significantly improve speech quality over such past techniques as the unconstrained Lim-Oppenheim technique as well as spectral subtraction with magnitude averaging [3]). Although significant improvement was noted, it was possible the algorithms were improving one or two particular speech classes which had high concentrations over the speech considered. Therefore, a comparative evaluation over speech sound classes was performed. Improvement over all classes of speech was reported.

As mentioned, the iterative enhancement algorithms must be suspended at some iteration. In order to determine a terminating iteration, a criterion must be selected to evaluate levels of improvement as the iterative scheme progresses. The criterion chosen is based on objective speech quality measures. Such measures are formed by a weighted comparison of actual and resulting estimated LPC predictor coefficients found during enhancement. The obvious problem with such a criterion is that, outside of simulation, the actual speech is unknown during the procedure. If, however, simulations were to show a consistent value for the best iteration in terms of this criterion, a convenient stopping condition would exist. Previous results based on objective quality measures indicate the unconstrained approach to produce maximum objective quality at different iterations for different classes of speech. Table 1 illustrates this behavior over the indicated sound classes. As this table shows, maximum overall speech quality is obtained at the third iteration, with considerable variation across sound types. For example, glides required two iterations, with nasals, liquids, and affricates requiring between five and six. Therefore, depending on sound class concentration, the optimal iteration (in terms of minimum distance) would vary considerably. This result indicates the inability to determine in advance a terminating iteration for the unconstrained approach since it is highly dependent on sound class and to a lesser degree on SNR.

The new constrained enhancement algorithms appear to solve this problem of sound class dependency. Table 2 presents results from an equivalent evaluation for one of the constrained enhancement algorithms (FF-LSP:T,Auto:I). A comparison between tables 1 and 2 show that the constrained approach produces superior quality measures across all speech classes at the same iteration. This improvement surpasses even combined individual maximum quality measures found across the unconstrained approach. Thus, the constrained enhancement algorithm does more than simply impose a constraint to adjust the rate of improvement: the constrained approaches consistently result in superior objective speech quality at the same iteration over all sound classes, independent of SNR. Table 3 summarizes optimum terminating points in terms of objective quality for the enhancement algorithms. Techniques employing only inter-frame constraints consistently resulted (93% occurrence) in maximum quality at the third iteration. Techniques employing inter- and intra-frame constraints had a 97% occurrence of maximum quality at the seventh iteration. In addition, adjacent iterations differ only slightly in objective quality for the constrained techniques. This is in sharp contrast to the large variations in adjacent iterations for the unconstrained technique. Therefore, if the iterative scheme were allowed to continue or halted one iteration

rior to optimal, only minor differences in speech quality would result. The results consistently suggested that the constrained enhancement algorithms reach a maximum level of speech quality at the same iteration, independent of SNR and sound class concentrations.

Sound Type	Itakura-Saito Likelihood Measure (across iterations)							
	Original	#1	#2	#3	#4	#5	#6	#7
Silence	1.63	1.62	1.61	1.65	1.93	3.76	20.36	49.80
Vowel	4.02	3.72	3.45	3.30	3.72	8.32	121.8	—
Nasal	19.81	19.15	18.42	17.06	17.01	16.59	15.19	15.70
Stop	7.36	6.11	4.93	3.98	3.82	6.89	25.52	29.69
Fricative	3.74	3.64	3.53	3.51	3.90	7.06	47.83	94.11
Glide	1.53	1.41	1.33	1.44	2.23	4.30	8.39	15.58
Liquid	9.60	8.24	6.55	4.55	3.61	1.68	6.38	30.00
Affricate	3.92	3.61	3.21	2.70	3.09	1.55	2.91	2.98
Voiced + Unvoiced	5.84	5.32	4.77	4.29	4.29	7.35	81.87	—
Total	4.02	3.72	3.40	3.15	3.27	5.80	43.46	—

Table 1: Lim-Oppenheim unconstrained speech enhancement for AWGN, SNR=+5dB. Optimum perceived quality or a particular speech class is indicated by a ♣.

Sound Type	Itakura-Saito Likelihood Measure (across iterations)								
	Original	#1	#2	#3	#4	#5	#6	#7	#8
Silence	1.63	1.55	1.35	1.16	1.03	0.98	0.93	0.88	0.90
Vowel	4.02	3.32	2.87	2.39	1.88	1.58	1.57	1.56	1.83
Nasal	19.81	16.49	12.40	10.52	8.88	6.84	4.93	3.79	5.55
Stop	7.36	6.25	4.84	3.49	2.67	1.81	1.38	1.13	1.43
Fricative	3.74	3.43	3.03	2.61	2.24	1.95	1.73	1.61	1.84
Glide	1.53	1.39	1.28	1.23	1.21	1.19	1.16	1.15	1.22
Liquid	9.60	6.48	3.38	2.24	1.61	1.21	0.94	0.92	1.21
Affricate	3.92	3.72	3.45	3.12	2.80	2.60	2.47	2.37	3.06
Voiced + Unvoiced	5.84	4.64	3.66	3.01	2.50	2.13	1.86	1.74	1.95
Total	4.02	3.03	2.44	2.07	1.80	1.61	1.46	1.38	1.49

Table 2: Hansen-Clements Inter & Intra-frame constrained speech enhancement for AWGN, SNR=+5dB. Optimum perceived quality for a particular speech class is indicated by a ♣.

Constrained Enhancement Algorithm	Additive White Gaussian Noise SNR								OVERALL	
	-5 dB		-0 dB		+5 dB		+10 dB			
	Optimal Iteration using Itakura-Saito Likelihood Measure									
	Iter.	Freq.	Iter.	Freq.	Iter.	Freq.	Iter.	Freq.		
FF-LSP:T	3	100%	3	87%	3	87%	3	100%	3	91%
			4	13%	4	13%			4	9%
VF-LSP:T	3	90%	3	85%	3	94%	3	100%	3	93%
	4	10%	4	15%	4	6%			4	7%
FF-LSP:T,Auto,I	7	100%	7	100%	7	100%	7	88%	7	97%
							8	12%	8	3%
FF-LSP:T,LSP,I	4	100%	4	100%	4	100%	4	100%	4	100%
VF-LSP:T,LSP,I	4	100%	4	100%	4	100%	4	100%	4	100%

Table 3: Summary of optimal terminating iteration across SNR for AWGN.

Additive Non-White, Non-Stationary Noise

The unconstrained Wiener filtering/all-pole modeling approach was previously generalized for colored aircraft noise [1]. In that study, an extensive investigation was performed using various spectral estimation techniques (MEM, MLM, Burg, Bartlett, Pisarenko, periodogram) for securing estimates of colored background noise, along with varying SNR (-20dB to +20dB). Results indicated that Bartlett's method produced spectral estimates which resulted in highest quality improvement for this particular distortion.

Noise recorded from a Lockheed C130 aircraft interior was used to degrade noise free utterances. For these simulations, two Bartlett spectral estimates found from the original noise waveform (to avoid complications in silence detection) were used across each sentence. The noise was both colored and non-stationary, so increasing the

number of spectral estimates across the utterance should improve enhancement performance. An analysis was performed for an inter-frame (FF-LSP:T), and a combined inter and intra-frame (FF-LSP:T,Auto:I) approach. Informal listening tests indicated noticeable quality improvement. Figure 2 illustrates results from this study. All configurations examined showed significant improvement in Itakura-Saito measures. Plot a shows Itakura-Saito measures for the original distorted speech. Plot b is from the unconstrained Wiener filtering technique. Plots c and d are typical values for the inter-frame constraint (FF-LSP:T), and inter- plus intra-frame constraint (FF-LSP:T, Auto:I) approaches. In order to determine limits on the level of enhancement, the original undistorted predictor coefficients were used in the unconstrained algorithm. In essence, the two step MAP estimation approach is now reduced to a single MAP estimate of \tilde{S}_0 , and therefore represents the theoretical limit for enhancement using Wiener filtering. Plot e indicates this limit. Although only Itakura-Saito measures are shown, similar improvement was observed for log area ratio and weighted spectral slope distance measures. As this figure indicates, significant levels of enhancement result for the constrained enhancement algorithms.

These results show that the constraint algorithms outperform the unconstrained approach for a colored distortion. However, it is possible that the constrained techniques are improving only particular speech classes which may have high concentrations in the test utterances. Therefore, a performance evaluation over sound classes was performed by hand partitioning speech into segments, pro-

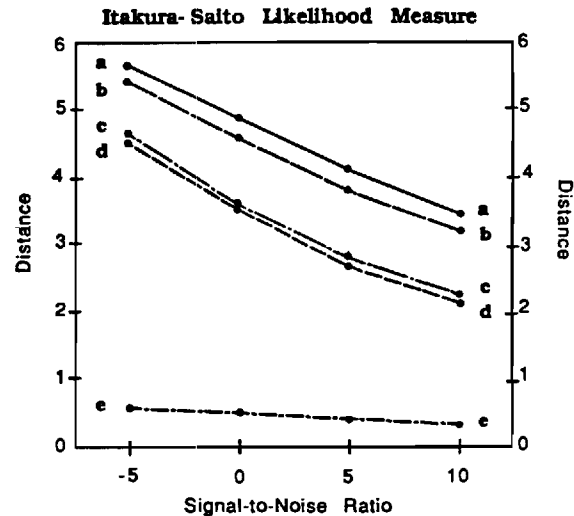


Figure 2: Comparison of inter & intra-frame constrained enhancement algorithms for colored aircraft noise over SNR.

- a.) Original Distorted Speech
- b.) Lim-Oppenheim: Unconstrained Wiener filtering
- c.) Hansen-Clements: employing Inter-Frame constraints
- d.) Hansen-Clements: employing Inter & Intra-Frame constraints
- e.) Theoretical limit: using undistorted LPC coefficients \tilde{a} .

ing entire sentences, and computing objective measures from each class. Table 4 summarizes this comparison between the unconstrained technique to that of the inter- and intra-frame constraint approach (FF-LSP:T,Auto:I). Measures for the theoretical limit using undistorted LPC coefficients are also indicated. It should be noted that voiced plus unvoiced measures give a better indication of quality improvement due to the time varying nature of the interfering background noise. Improvement is indicated for all types of speech. This shows that the constrained techniques are enhancing all aspects of the speech signal.

Sound Type	Itakura-Saito Likelihood Measure			
	Original	Lim-Oppenheim	Hansen-Clements	True LPC
Silence	6.63	6.33	4.32	2.03
Vowel	3.23	2.54	1.44	0.53
Nasal	4.03	3.26	2.13	0.45
Stop	1.58	1.29	0.66	0.61
Fricative	1.37	1.09	0.85	0.65
Glide	1.14	1.04	0.52	0.51
Liquid	1.22	0.55	0.22	0.18
Affricate	0.90	0.51	0.33	0.16
Voiced + Unvoiced	2.27	1.76	1.08	0.52
Total	4.15	3.86	2.74	1.17

Table 4: Comparison of unconstrained (Lim-Oppenheim) and inter- and intra-frame constrained (Hansen-Clements) algorithms over sound types for slowly varying colored noise. SNR = +5 dB

Recognition Evaluation

A fairly standard, isolated-word, discrete-observation hidden Markov model recognition system was used for evaluation. This system was LPC based and had no embellishments. In all experiments, a five state, left-to-right model was used. System dictionary consisted of twenty highly confusable words used by Texas Instruments and Lincoln Labs to evaluate recognition systems. Subsets include {go,oh,no,hello} and {six,fix}. Twelve examples of each word were used, six for training, six for recognition (i.e., all tests fully open). A vector quantizer was used to generate a 64 state codebook using two minutes of noise free training data. The twenty models employed by the HMM recognizer were trained using the forward-backward algorithm. Table 5 presents results from five scenarios using a noise free codebook and noise free trained system. Spectral subtraction preprocessing employed three frames of magnitude averaging. The unconstrained Lim-Oppenheim approach was terminated at the third iteration. The constrained Hansen-Clements (FF-LSP:T,Auto:I) was terminated at the seventh. As these results indicate, recognition was reduced to chance for noisy, spectral subtraction, and Lim-Oppenheim (-5,0,5 dB) speech. The constrained approach resulted in improved recognition across all SNR considered, which is quite remarkably in light of the severe levels of noise, and difficulty of dictionary employed. However, reliable recognition in such a hostile environment may require more than merely extending existing techniques. As a final comparison, three tests were performed using noisy and enhanced speech (SNR=+10dB). For the noisy case, speech was coded using a noisy codebook, and recognition performed using a noisy trained HMM recognizer. Similar tests were performed for two enhancement techniques, (i.e., enhanced words coded using enhanced codebook, and tested using enhanced speech trained HMM recognizer). 40% of the errors in recognition were caused by misclassification of leading consonants (especially fricatives).

Condition (noise free training)	RECOGNITION RESULTS Signal-to-Noise Ratio				
	Original	-5dB	0dB	+5dB	+10dB
Noise free	88%				
Noisy		5%	5%	6.7%	5%
Spectral Subtraction		5.8%	7.1%	5%	5.4%
Lim-Oppenheim		5.4%	5.8%	7.5%	12.5%
Hansen-Clements		15%	14%	19.5%	34.5%
Train & Recognize In Same Environment					
Noise free	Noisy ↑	Hansen-Clements ↑		Lim-Oppenheim ↑	
88%	90%	77%		23%	

Table 5: Recognition performance using enhancement preprocessing in AWGN. ↑ SNR = +10dB

5 Conclusions

The constrained speech enhancement algorithms have been shown to improve speech quality across all classes of speech for both additive white Gaussian and slowly varying, non-white degradations. In addition, a consistent terminating procedure has been identified which is independent of sound class concentration and relatively insensitive to varying SNR. Finally, the constrained algorithms have shown improvement as a preprocessor for speech recognition, although their ability to bring performance up to an acceptable level in SNR's low as those considered is questionable. Though the enhancement procedures improved LPC parameter estimation substantially, LPC-based strategies may simply be inappropriate for SNR's of roughly 0dB. Further work in this SNR range will require as a minimum, different front end processing.

This work sponsored in part by U.S. Army Human Engineering Labs.

References

- [1] J.H.L. Hansen, M.A. Clements, "Enhancement of Speech Degraded by Non-White Additive Noise," Final Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [2] J.H.L. Hansen, M.A. Clements, "Objective Quality Measures Applied to Enhanced Speech," *Proc. of the Acoustical Society of America*, 110th Meeting, C11, Nashville, Tenn., Nov. 1985.
- [3] J.H.L. Hansen, M.A. Clements, "Iterative Speech Enhancement With Spectral Constraints," *Proc. 1987 IEEE ICASSP*, pp. 189-192, Dallas, TX, April 1987.
- [4] J.H.L. Hansen, M.A. Clements, "Constrained Iterative Speech Enhancement," *IEEE Trans. on Acoust., Speech, Signal Processing*, submitted, Dec. 1987.
- [5] J.S. Lim, A.V. Oppenheim, "All-Pole Modeling of Degraded Speech," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, pp. 197-210, June 1978.
- [6] S.R. Quackenbush, "Objective Measures of Speech Quality," Ph.D. Thesis, Georgia Institute of Technology, May 1985.

Constrained Iterative Speech Enhancement

by

John H. L. Hansen⁴, Member, IEEE

and

Mark A. Clements⁵, Member, IEEE

June 29, 1989

¹Submitted March 20, 1988, Revised June 21, 1989

²This work sponsored in part by grants from U.S. Army Human Engineering Labs and Department of Defense.

³Please address all correspondence to Dr. Hansen.

⁴J.H.L. Hansen was with the School of Electrical Engineering, Georgia Institute of Technology, Atlanta, GA. He is now with the Department of Electrical Engineering, Duke University, Durham, NC 27706.

⁵M.A. Clements is with the School of Electrical Engineering, Georgia Institute of Technology, Atlanta, GA 30332.

Abstract

In this paper, an improved form of iterative speech enhancement for single channel inputs is formulated. The basis of the procedure is sequential maximum a posteriori estimation of the speech waveform and its all-pole parameters as originally formulated by Lim and Oppenheim, followed by imposition of constraints upon the sequence of speech spectra. The new approaches impose intra- and inter-frame constraints on the input speech signal to ensure more speech-like formant trajectories, reduce frame-to-frame pole jitter and effectively introduce a relaxation parameter to the iterative scheme. Recently discovered properties of the line spectral pair representation of speech allow for an efficient and direct procedure for application of many of the constraint requirements. Substantial improvement over the unconstrained method has been observed in a variety of domains. First, informed listener quality evaluation tests and objective speech quality measures demonstrate the technique's effectiveness for additive white Gaussian noise. A consistent terminating point for the iterative technique is also shown. Second, the algorithms have been generalized and successfully tested for noise which is non-white and slowly varying in characteristics. The current systems result in substantially improved speech quality and LPC parameter estimation in this context with only a minor increase in computational requirements. Third, the algorithms were evaluated with respect to improving automatic recognition of speech in the presence of additive noise, and shown to outperform other enhancement methods in this application.

1 Introduction

The presence of background noise can seriously degrade the performance of many speech processing systems, since most digital voice communication and recognition systems have traditionally been formulated in noise-free, tranquil environments. There are, however, many instances where such systems must perform reliably in noisy environments. As an example, consider the use of speech recognition in a noisy aircraft cockpit. It has been shown that recognition performance is severely reduced in such an environment due to background noise and pilot task requirements [8, 13, 18]. Since commonly used front-ends do not usually take noise into account explicitly, recognition deteriorates rapidly. One alternative, which would benefit recognition as well as speech coding systems is to develop enhancement preprocessors that produce speech or recognition features less sensitive to background noise, so that existing recognition/communication systems may be employed. Such preprocessing systems would also benefit human listeners by improving speech characteristics in voice communications systems.

The problem of enhancing speech degraded by additive background noise covers a broad spectrum of applications and issues [12]. A system may be directed at one or more objectives such as improving overall quality, increasing intelligibility, or reducing listener fatigue. Assumptions made in this investigation include: i) the background noise distortion is additive, ii) only the degraded speech signal is available (i.e., single microphone environment), and iii) the noise and speech signals are uncorrelated.

This paper presents an improved method for iterative speech enhancement based on a set of vocal tract spectral constraints. The framework of this approach was adopted from all-pole modeling/noncausal Wiener filtering as formulated by Lim and Oppenheim [11]. The original iterative technique attempts to solve for the maximum a posteriori (MAP) estimate of a speech waveform in additive white noise. The improved techniques are formulated using inter- and intra-frame constraints to ensure speech-like characteristics. An efficient technique for applying the spectral constraints is based on the line spectral pair (LSP) transformation of the LPC parameters. The paper is arranged as follows. First, the iterative unconstrained technique is discussed. Several anomalies are cited which

motivate formulation of constrained enhancement techniques using the LSP transformation. Next, algorithm evaluation is performed for additive white Gaussian noise, and a slowly varying non-white distortion. Finally, a comparative evaluation is also performed to determine their usefulness as preprocessors for recognition in noisy environments.

2 Iterative Speech Enhancement

Enhancement based on the estimation of all-pole speech parameters in additive white Gaussian noise was investigated by Lim and Oppenheim [11], and later for a colored noise degradation by Hansen and Clements [3, 4, 6]. This approach attempts to solve for the maximum a posteriori estimate of a speech waveform in additive white Gaussian noise with the requirement that the signal be the response from an all-pole process. Crucial to the success of this approach is the accuracy of the estimates of the all-pole parameters at each iteration. After some simplification, it can be shown that the resulting equations for the joint MAP estimate of the all-pole speech parameters \vec{a} , gain g , and noise free speech \vec{S}_O become nonlinear. Lim and Oppenheim considered a suboptimal solution employing sequential MAP estimation of \vec{S}_O followed by MAP estimation of \vec{a} , g given $\hat{\vec{S}}_{O,i}$, where $\hat{\vec{S}}_{O,i}$ is the result of the i th estimation. The sequential estimation procedure is linear at each iteration, and must continue until some criterion is satisfied. With further simplifying assumptions, it can be shown that MAP estimation of \vec{S}_O is equivalent to noncausal Wiener filtering of the noisy speech \vec{Y}_O . Lim and Oppenheim showed this technique, under certain conditions, increases the joint likelihood of \vec{a} and \vec{S}_O with each iteration. It can also be shown to be the optimal solution in the mean-squared sense for a white noise distortion.

Although successful in a mathematical sense, this technique has received little application due to several factors. First, the scheme is iterative with sizable computational requirements. Second and most important, is that although the original sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, a heuristic convergence criterion had to be employed. This represents a serious drawback if the approach is to be used in environments requiring

automatic speech enhancement. Hansen and Clements performed an extensive investigation of this technique for additive white Gaussian (AWGN), and a generalized version for additive non-white, non-stationary aircraft interior noise [3, 4]. Objective speech quality measures, which have been shown to be correlated with subjective quality [17], were used in the evaluation. This approach was found to produce significant levels of enhancement for white Gaussian noise in 3-4 iterations. Improved all-pole parameter estimation was also observed in terms of reduced mean squared error. Only if the probability density function is unimodal, and the initial estimate for $\tilde{\mathbf{a}}$ is such that the local maximum equals the global maximum, is the procedure equivalent to the joint MAP estimate of $\tilde{\mathbf{a}}$, g and \tilde{S}_O . Some interesting anomalies were noted which helped motivate development of the constrained approaches. First, as additional iterations were performed, individual formants of the speech consistently decreased in bandwidth and shifted in location as indicated in Figure 1. Second, frame-to-frame pole jitter was observed across time. Both effects contributed to unnatural sounding speech. Third, although the sequential MAP estimation technique was shown to increase the joint likelihood of the speech waveform and all-pole parameters, a heuristic convergence criterion had to be employed. Lim and Oppenheim recognized these limitations and an improved method was formulated by Musicus and Lim [15] which addresses some of them. Even with their improvements, however, no explicit frame-to-frame constraints are employed. Since the original algorithm already constrains the speech to be the response from an all-pole system, applying further constraints on the pole movements imposes no new assumptions on the speech or noise, and may improve the algorithm’s performance. The imposition of some relatively simple constraints turns out to improve speech quality results, even when directly attached to the original Lim-Oppenheim method.

Enhancement with Spectral Constraints

Consider the statistical parameter estimation of speech in the presence of noise as formulated by Lim and Oppenheim where all unknown parameters over a short interval (all-pole speech parameters $\tilde{\mathbf{a}}$, gain g , and noise free speech \tilde{S}_O) are random with a priori Gaussian probability density functions. It was shown that MAP estimation of $\tilde{\mathbf{a}}$, g , and

\vec{S}_O given noisy observations \vec{Y}_O , results in a set of nonlinear equations. Therefore, instead of joint estimation of \vec{a} and \vec{S}_O , a suboptimal solution was formulated employing a two-step approach based on MAP estimation of \vec{S}_O given \vec{Y}_O , followed by MAP estimation of \vec{a} , g given $\hat{\vec{S}}_{O,i}$, where $\hat{\vec{S}}_{O,i}$ is the result of the i th estimation. In the currently reported work, constraints are imposed on the vocal tract spectrum between MAP estimation steps. The procedure for obtaining the MAP estimates of \vec{a} and g remain the same, as that of Lim and Oppenheim. In the current system, constraints are applied to $\hat{\vec{a}}_i$ to ensure that, i) the all-pole speech model is stable, ii) it possesses speech-like characteristics (e.g., poles are in reasonable places with respect to each other and the unit circle), and iii) the vocal tract characteristics do not vary by more than a prescribed amount from frame to frame when speech is present. Given the new estimate $\hat{\vec{a}}_{i+1}$, the second MAP estimation of \vec{S}_O is performed by maximizing its conditional probability density function given $\hat{\vec{a}}_{i+1}$ and the observed noisy sequence \vec{Y}_O . Since this probability density function is jointly Gaussian, the resulting MAP estimate is equivalent to a MMSE estimate of \vec{S}_O . With further simplifying assumptions, it can be shown that MAP estimation of \vec{S}_O reduces to a minimum mean squared error (MMSE) estimate, and as the observation window increases, the procedure becomes a noncausal Wiener filter. Once the new estimate of $\hat{\vec{S}}_{O,i}$ is formed, the iterative procedure continues by re-estimating $\hat{\vec{a}}_i$, applying constraints to $\hat{\vec{a}}_i$, and forming the noncausal filter using $\hat{\vec{a}}_{i+1}$ to re-estimate $\hat{\vec{S}}_{O,i}$. The procedure continues until some convergence criterion is satisfied. Due to the flexibility of the enhancement framework, a variety of constraint options are possible between MAP estimation steps.

Figure 2 presents an overview of two classes of constraints which include inter-frame (across time) and/or intra-frame (across iterations). Each technique differs in the type of constraint and computational requirements. The present evaluation focuses on two representative inter-frame (FF-LSP:T) and combined inter-frame plus intra-frame (FF-LSP:T,Auto:I) based techniques. Further discussion of all techniques are found in [5, 6, 7]. For historical purposes, several comments concerning the other approaches are summarized.

Since observations indicate that poles of the LPC filter often move unrealistically close to the unit circle when the unconstrained iterative technique is allowed to continue,

initial techniques limited pole movement by applying constraints directly to radial and/or angular movements of the LPC poles across iterations and time. For these techniques, LPC predictor coefficients were obtained, a P th-order root-solve was performed and a pole ordering step applied. If pole movement fell within a movement constraint window, a constraint was applied, otherwise, no constraint was applied based on the assumption that either movement was allowable, or that the pole was mischaracterized due to the ordering step. Results showed substantial improvement in objective speech quality (as measured by Itakura-Saito, log-area-ratio, and weighted spectral slope (Klatt) measures [17]). Informal listening tests also revealed improvement, especially during vowels and vowel transitions toward nasals. Larger levels of quality improvement were observed using inter-frame versus intra-frame constraints, thus suggesting that temporal variation in pole locations have a greater effect on overall quality.

Although successful in improving speech quality, constrained techniques based on direct pole location were computationally expensive. A P th-order root-solve and a pole ordering step per frame for each iteration was required. Since root solving is not always numerically accurate and ordering can be inconsistent across frames, a more robust approach was sought to implement these constraints.

An alternative approach for implementing the spectral constraints was formed by employing the line spectral pair (LSP) transformation as a method for representing the vocal tract spectrum. Previous success of the LSP transformation in low-bit-rate speech coding by Crosmer [2] led to the use of LSP's for this purpose.

The Line Spectral Pair (LSP) [9, 19] transformation comes from modifying the LPC polynomial, $A(z)$, in two ways: $P(z)$ and $Q(z)$ are obtained by augmenting $A(z)$'s *PAR-COR* sequence with a $+1$ and -1 respectively. This results in two polynomials of order $p + 1$ which have all roots on the unit circle.

$$P(z) = (1 - z^{-1}) \prod_{i=1,3,5,\dots}^{M-1} (1 - 2 \cos \omega_i z^{-1} + z^{-2}) \quad (1)$$

$$Q(z) = (1 + z^{-1}) \prod_{i=2,4,6,\dots}^{M-1} (1 - 2 \cos \omega_i z^{-1} + z^{-2}) \quad (2)$$

The angles of the roots, $\{\omega_i, i = 1, 2, \dots, M\}$, are called the *line spectrum pairs*. In general, $A(z)$ will represent a stable LPC filter if and only if the roots of $P(z)$ and $Q(z)$

interleave. The angles of the roots of $P(z)$, correspond roughly to the angles of the roots of $A(z)$ (formant frequencies), and the separation of a particular root of $P(z)$ from the closest root of $Q(z)$ indicates in some sense the bandwidth of that resonance. The angle of the roots of $P(z)$ between 0 and π are termed the position parameters (i.e., the odd indexed LSP parameters, $\{p_i = \omega_{2i-1}, i = 1, 2, \dots, M/2\}$), and the separations mentioned above are the difference parameters, d_i .

$$\{|d_i| = \min_{j=-1,1} (|\omega_{2i+j} - \omega_{2i}|), i = 1, 2, \dots, M/2\} \quad (3)$$

The sign of d_i is positive if ω_{2i} is closer to ω_{2i+j} , and otherwise is negative. The useful properties of the LSP's include an easy check for stability, excellent interpolation properties, ease of computation (compared to roots of $A(z)$), some well understood trajectories for speech, and the relative insensitivity of the auditory system under quantization of the difference parameters.

Enhancement Using the LSP Transformation

In these techniques, constraints are imposed on the LSP parameters directly. In the first technique (MS-LSP:T), a five frame median smoothing constraint was placed on the position parameters across time, with difference parameters restricted to be at least d_{MIN} in magnitude, ensuring the LPC poles of reasonable bandwidth. Good improvement resulted without the expense of root solving or pole ordering. Plots of LSP parameters versus time confirmed a reduction in frame-to-frame pole jitter with only a slight increase in computational requirements. Since vocal-tract characteristics and relative strength of background noise vary across time, the imposition of spectral constraints should be dependent on speech characteristics obtained during the enhancement procedure. Therefore, the remaining constraints are applied based on particular characteristics found in the speech waveform during enhancement.

Two inter-frame approaches are considered: a fixed frame rate (FF-LSP:T), and a variable frame rate approach (VF-LSP:T). In the first of these, the LPC predictor coefficients, \vec{a} , are first converted to LSP parameters. Next, each frame's energy is observed, and classified as voiced or unvoiced speech according to some threshold $E_{V/UV}$. A local

running count L_i is kept for the number of consecutive frames which fall below the energy threshold. If L_i reaches L_{MAX} , all subsequent frames below the threshold are classified as noise. This allows for a tighter pole movement constraint during long periods of silence. The position parameters for each frame are smoothed using a weighted triangular window with a variable base of support (1 to 5 frames). If a frame has been classified as noise, maximum smoothing (or tightest movement constraint) is performed. The lower formant frequencies are smoothed over a narrower triangle width than for those position parameters at higher frequencies in order to preserve perceptually important speech characteristics found in the lower formants. No smoothing is performed on the difference parameters since they are more closely related to formant bandwidth than formant location. However, it is possible that a difference parameter falls within a "forbidden zone." When this occurs, the LPC analysis has most likely underestimated a particular pole's bandwidth. Since this causes unnatural sounding speech, (as found in the unconstrained approach), the value of $|d_i|$ is set to d_{MIN} . Finally, the position and difference parameters are combined to form the constrained LPC predictor coefficients \hat{a}_{i+1} .

The (FF-LSP:T) technique applies constraints across time on a frame-by-frame basis. Since phonetic transitions do not normally coincide with frame boundaries, an inter-frame approach (VF-LSP:T) based on constraints applied over speech segments was formulated. The technique is identical in theory to (FF-LSP:T), except for the front-end segmentation algorithm which divides the signal into speech segments. Segments are chosen to be long when the speech spectrum is slowly varying and short when the speech spectrum is varying quickly. The LSP parameters are reconstructed with linear interpolation used to compute the parameters for intermediate frames.

The segmentation algorithm begins by determining the onset/offset of speech by thresholding the LPC residual energy, which produces relatively long segments. Long segments are subdivided based on the curvature of the position parameters. This is performed by computing a gain-normalized Itakura-Saito measure of the spectral distance between the frequency response of two adjacent frames. The procedure continues by computing spectral distortion of position parameters for successively longer segments until the spectral distortion exceeds a threshold T_D . At that point, a subsegment boundary

is set, with the intermediate position parameters reconstructed via linear interpolation. During this step, the length of a subsegment is also limited to L_{MAX} to prevent excessively long segments which might contribute to muffled or unnatural sounding speech. The advantage of this approach is to incorporate more information from adjacent frames when the spectrum indicates similar characteristics. This in effect, distorts the position parameters as little as possible when associated difference parameters indicate the presence of formants. Difference parameters for each frame are used to compute the predictor coefficients \hat{a}_{i+1} . The difference parameters are required to be at least d_{MIN} or greater.

The convergence problems inherent in the unconstrained Wiener filtering approach which have been pointed out [5, 7, 15], are at least partially caused by bias in the MAP estimation. Although spectral constraints were originally constructed to be used across frames, it has been observed that if they are used across iterations, convergence to reasonable values occurs with much greater frequency and consistency. In particular, previous results based on objective speech quality measures show the unconstrained Wiener filtering approach to produce minimum objective measures at different iterations for different classes of speech [5, 7] (see Table 3). By constraining the vocal tract filter to be a function of its values obtained from previous iterations, a much improved consistency in quality across speech classes and LPC parameter \hat{a}_i estimation resulted. Two approaches were considered, one applied to the autocorrelation lags (Auto:I), the other to the position parameters (LSP:I). The first approach simply weighted the present set of autocorrelation lags with the same frame from previous iterations. Such a technique is easy to perform, since the autocorrelation lags must be computed in order to estimate the predictor coefficients \vec{a} . The second approach weighted position parameters with those from the same frame but previous iteration. If the corresponding difference parameter indicated the adjacent position parameter to represent a formant, this approach had the effect of constraining the formants to lie along smooth tracks across iterations. Such a procedure is generally referred to as introducing relaxation into the iterations [16]. If the iteration is producing results for which weighted averaging makes sense (e.g., LSP's but not \vec{a}), improved convergence results. Results from inter-, intra-, and combined inter-plus intra-frame constraint approaches will be presented in the next section. Figure 3

illustrates the framework for the new set of constrained enhancement techniques.

3 Evaluation

We now evaluate the performance of the proposed algorithms for speech enhancement alone, and as a preprocessor for word recognition in noisy environments. Speech was degraded by additive white or colored noise and processed. Enhancement algorithms evaluated include: techniques incorporating inter-frame constraints applied on a fixed-frame (FF-LSP:T) or variable-frame (VF-LSP:T) basis to the LSP parameters, and algorithms incorporating combinations of inter- plus intra-frame constraints (FF-LSP:T,Auto:I), (FF-LSP:T,LSP:I). Global estimates of SNR¹ were used in the evaluation, since the assumption of accurate local estimates is normally unrealistic in actual noisy environments. Further improvement is therefore possible if a continuous local SNR estimate is available. The Intra-frame constraints were applied across two to three iterations.

Several parameters must be addressed to ensure proper application of spectral constraints. These include the voiced/unvoiced energy threshold $E_{V/UV}$, silence frame count threshold L_{MAX} , LSP difference parameter thresholds d_{MIN} , d_{MAX} , and the accumulated frame-to-frame Itakura-Saito distance threshold T_D .

The energy threshold $E_{V/UV}$ is used to distinguish voiced from unvoiced or silent speech frames for use in applying inter-frame constraints. Values were obtained from frame energy histograms at each signal-to-noise ratio. Similar enhancement levels resulted for $E_{V/UV}$ in the range between average, and one standard deviation below average speech frame energy (e.g., Average frame energy for sentence S6 was 7719. $E_{V/UV}$ set between 8000 and 5000 resulted in Itakura-Saito measures which ranged from 1.96 to 2.02).

The silence frame count threshold L_{MAX} , is used in conjunction with $E_{V/UV}$. If L_{MAX} consecutive frames fall below $E_{V/UV}$, that segment is classified as silence (or noise) so that tighter spectral constraints can be enforced. If $E_{V/UV}$ is set as above, similar speech

¹The signal-to-noise ratio is defined as $10 \log \left(\frac{\sum_n s^2(n)}{\sum_n d^2(n)} \right)$, where the summation is over the entire length of the sentence. This definition was chosen in keeping with the format used in previous studies on noncausal Wiener filtering. [11]

quality measures resulted with L_{MAX} set between two and five frames. Reduced quality measures resulted with L_{MAX} in the eight to twelve frame range, thereby suggesting increased residual noise levels during silent portions.

The difference thresholds d_{MIN}, d_{MAX} , constrains the LSP difference parameters to ensure poles of reasonable bandwidths (e.g., the all-pole speech model is stable and that it possesses speech-like characteristics). Values in the range $.015 \leq d_{MIN} \leq .031$ radians, $.055 \leq d_{MAX} \leq .077$ radians, resulted in good quality improvement.

The value of T_D (accumulated frame-to-frame Itakura-Saito distance threshold) greatly effects speech segment length. If set to high, small duration phonemes can be lost (e.g., an initial stop and final vowel joined to form one speech segment as in *be*). A value of 1.2 was found to produce segments of reasonable length and quality at higher SNR ($\geq +5\text{dB}$). At lower SNR, frame-to-frame distance values were too large to reliably segment speech, resulting in decreased performance.

Generally speaking, substantial enhancement resulted for a wide range of E_V/U_V , L_{MAX} , d_{MIN} , and d_{MAX} threshold settings, indicating the algorithms robust performance over estimated threshold values. Only T_D , the accumulated frame-to-frame Itakura-Saito distance threshold, proved to be sensitive, especially across varying SNR. Greater enhancement was observed when T_D was allowed to vary across iterations.

In this study, the primary tool for quantitative enhancement evaluation has been objective quality measures. This is based on extensive work carried out in the formulation of objective speech quality measures for speech coding [1.7], and the application of these measures to enhancement [4]. Fair to good correlation has been shown to exist between subjective and objective quality measures, such as: the Itakura-Saito likelihood ratio, log area ratio, and weighted spectral slope measure. These measures have been shown to be a viable tool for use in evaluating speech enhancement algorithms for white and non-white additive noise [4]. In addition, the Itakura-Saito likelihood ratio is also a commonly used distance measure for speech recognition as well as for coding methods employing vector quantization. Therefore, improvement in Itakura-Saito distance might also suggest the possibility of improvement in automatic recognition. The speech data for enhancement evaluation is described in the Appendix.

3.1 Evaluation Using Additive White Gaussian Noise

Various configurations of the new constrained enhancement algorithms were evaluated in an additive white Gaussian noise environment. Informal listening tests indicated noticeable quality improvement, although no intelligibility testing was performed. A variety of objective speech quality measures were used in the evaluation procedure. Figure 4 illustrates a comparison of typical results for the various constraint approaches. The Itakura-Saito measure is plotted versus signal-to-noise ratio for a white noise distortion. Plot *a* represents the original distorted speech. Plots *b* through *e* represent combinations of inter-frame constraints (both fixed and variable rate), and intra-frame constraints (applied to position parameters/autocorrelation lags). All configurations examined showed significant improvement in Itakura-Saito measures. Threshold settings for the variable frame rate inter-frame constraint were somewhat sensitive to varying noise levels. This indicates that although applying inter-frame constraints across speech segments is theoretically attractive and should aid in enhancement, in reality the speech segmentation step proves to be too sensitive to varying background noise levels. However, the fixed frame approach by itself, and with either autocorrelation or position intra-frame constraints gave impressive results with little sensitivity to varying levels of SNR. In order to determine a limit on the level of enhancement, the original undistorted predictor coefficients \vec{a} were used in the unconstrained algorithm. In essence, the two step MAP estimation approach is now reduced to a single MAP estimate of \vec{S}_O , and therefore represents the theoretical limit for enhancement using Wiener filtering. Plot *f* indicates this limit.

One advantage of the general class of Wiener filtering approaches is that no “musical tone” artifacts are present after processing as observed in spectral subtraction techniques[1, 3, 12]. To determine performance versus spectral subtraction, a series of enhancement evaluations under identical conditions (same distorted utterances, same global SNR estimates) were performed. Evaluation was performed for both half and full-wave rectification over a SNR range of -20 to $+20$ dB, and employed one to five frames of magnitude averaging (as defined by Boll [1]). See Hansen [7] for details. Full-wave rectification resulted in improvement over a wider range of SNR, however half-wave recti-

fication had greater improvement over the restricted SNR band of 5 to 10 dB. Magnitude averaging lead to improved enhancement for both rectification approaches.

Next, the constraint approaches were compared to spectral subtraction and unconstrained noncausal Wiener filtering. All systems performed enhancement on the same speech, with the same global estimates of SNR. Figure 5 compares quality improvement for each technique. Although only Itakura-Saito measures are shown, similar improvement was observed for log area ratios and weighted spectral slope measures (Klatt). Itakura-Saito measures are presented since they are widely accepted as a spectral distance measure and have been used extensively for speech recognition applications. A comparison of the three speech quality measures is shown in Table 2. The average correlation between each objective quality measure and subjective quality as measured by the DAM (diagnostic acceptability test) is shown [17].

Quality Improvement Over Speech Classes

To determine individual quality improvement, an evaluation over sound classes was performed by hand partitioning speech into segments, processing entire sentences, and computing objective measures from each class. Table 1 summarizes the comparison between the unconstrained technique, and an inter- plus intra-frame constrained approach (FF-LSP:T,Auto:I). Measures for the theoretical limit using undistorted LPC predictor coefficients \tilde{a} are also indicated. Improvement is indicated for all classes of speech. These results show that the constraint techniques are enhancing all aspects of the speech signal.

Termination Criterion

As mentioned, the iterative enhancement algorithms must be suspended at some iteration. In order to determine a terminating iteration, a criterion must be selected to evaluate levels of improvement as the iterative scheme progresses. The criterion chosen is based on objective speech quality measures. Such measures are formed by a weighted comparison of actual and resulting estimated LPC predictor coefficients found during enhancement. The obvious problem with such a criterion is that, outside of simulation,

the actual speech is unknown during the procedure. If, however, simulations were to show a consistent value for the best iteration in terms of this criterion, a convenient stopping condition would exist. Previous results based on objective quality measures indicate the unconstrained approach to produce maximum objective quality at different iterations for different classes of speech. Table 3 illustrates this behavior over the indicated sound classes. As shown, maximum overall speech quality is obtained at the third iteration, with considerable variation across sound types. Glides required two iterations for maximum quality, with nasals, liquids, and affricates requiring between five and six. Therefore, depending on sound class concentration, the optimal iteration (in terms of minimum distance) would vary considerably. Observations from a previous investigation indicate that the optimal iteration varies between the second and sixth and that it is also somewhat dependent on SNR [3].

The new constrained enhancement algorithms have less sensitivity to sound class. Table 4 presents results from an equivalent evaluation for one of the constrained enhancement algorithms (FF-LSP:T,Auto:I). A comparison between tables 3 and 4 show that the constrained approach produces superior quality measures across all speech classes at the same iteration. The improvement surpasses even combined individual maximum quality measures found across the unconstrained approach. Thus, the constrained enhancement algorithm does more than simply impose a constraint to adjust the rate of improvement: the constrained approaches consistently result in superior objective speech quality at the same iteration over all sound classes, independent of SNR.

Termination Consistency Versus SNR

Further evaluations were performed to determine the consistency of the terminating iteration versus SNR. Table 5 summarizes optimum terminating points in terms of objective quality for some of the enhancement algorithms. Techniques employing only inter-frame constraints consistently resulted (94% occurrence) in maximum quality at the third iteration. Techniques employing inter- and intra-frame constraints had a 97% occurrence of maximum quality at the seventh iteration. In addition, due to the relaxation of the iterative scheme as imposed by intra-frame constraints, adjacent iterations differ

only slightly in objective quality for the constrained techniques. Therefore, only minor differences in speech quality would result if the iterative scheme were halted one iteration prior to optimum. The results consistently suggest that the constrained enhancement algorithms reach a maximum level of speech quality at the same iteration, independent of SNR and sound class concentrations. Thus, a convenient terminating criterion may be determined under simulated conditions and employed in actual noisy environments.

Vocal Tract Estimation

In addition to the problem of a terminating point dependent on speech class concentration and SNR, the unconstrained approach also suffered from undesirable movements of the LPC poles. Specifically, it was observed that as additional iterations were performed, individual formants of the speech consistently decreased in bandwidth and shifted in location as shown in Figure 1. Figure 6 illustrates results from a single frame of speech for the unconstrained and constrained approaches. The *original* and *distorted original* spectra are the same for both approaches. Results from *4 iterations* and *8 iterations* are presented for both approaches. For the unconstrained approach, the terminating point is the fourth iteration. For this example the unconstrained approach was somewhat successful in improving overall spectral shape, especially in the region of the second formant. However, as additional iterations were performed, spectral distortions result, especially with respect to bandwidth information. The constraint approach (FF-LSP:T,Auto:I) is able to eliminate these undesirable effects. The terminating point for this approach was the seventh iteration. The change in spectral shape between the seventh and eighth iterations were minor, based on visual observation and objective speech quality measures. As this figure indicates, fine characteristics of the speech spectrum result only in the later iterations.

Computational Issues

Discussion of algorithm performance should also address computational issues as well as algorithm complexity. Naturally, there exists a trade-off between resulting speech

quality and each algorithm's computational complexity. It is clear that iterative techniques require greater computer resources than non-iterative approaches such as spectral subtraction and correlation subtraction. However, improvement in speech quality for the constraint approaches may be substantial enough to justify the additional computer requirements. In Table 6, a comparison of enhancement algorithms are made with respect to speech quality, relative computer resources and memory requirements, and algorithm complexity. By applying constraints to the LSP parameters, a modest increase in computer resources results in a marked increase in speech quality. For example, median smoothing of the LSP parameters (MS-LSP:T) increases speech quality with only slight increases in computation and complexity. If greater resources are available, more sophisticated constraint approaches may be chosen. If memory and computational resources are available, use of the constrained approaches appears justifiable.

Time Versus Frequency Plots

Isometric plots of time versus frequency magnitude spectra were constructed. In Figure 7, each line represents a 128-point frequency analysis. The top two graphs are the original and distorted cases. The lower left graph is the time versus frequency response for the unconstrained approach, terminated at the third iteration. The lower right graph is the frequency response after six iterations of an inter- plus intra-frame constrained (FF-LSP:T,Auto:I) approach. These figures indicate that the considerable noise rejection achieved in the single frame noted in Figure 6, is generally true over time.

3.2 Evaluation Using Additive Non-White, Non-Stationary Noise

The enhancement techniques described for the white additive noise case were also tested using non-stationary, colored noise recorded from the interior of a Lockheed C130 aircraft. Estimates for the noise spectrum were made using Bartlett's method [10, 14] over long

intervals². Energy thresholds for the inter-frame constraints were obtained from frame energy histograms at each signal-to-noise ratio. Intra-frame constraints were applied across two to three iterations. Figure 8 and Table 7 list the results of the analysis, presented in a manner consistent with the white noise descriptions. Although only Itakura-Saito measures are shown, similar improvement was observed for log-area-ratio and weighted spectral slope distance measures [7]. As seen, consistent improvement over all SNR's and speech sounds resulted, although the improvement was not as much as the white noise case.

3.3 Recognition Evaluation

One application for speech enhancement is a preprocessor for an automatic recognition system. For evaluation of the enhancement algorithms in this application, a set of recognition experiments were performed, including: 1) the no noise condition (in order to set an upper limit of recognition performance), 2) distorted condition with no preprocessing (in order to set an assumed lower limit of recognition), 3) the best performing spectral subtraction preprocessing (i.e., the configuration employing either half or full-wave rectification and 1 to 5 frames of magnitude averaging which gave the highest quality improvement for the given vocabulary), 4) unconstrained Lim-Oppenheim preprocessing, 5) and constrained preprocessing. The evaluation was performed at six levels of SNR (-5,0,+5,+10,+20,+30 dB) for the additive white Gaussian noise degradation.

A fairly standard, isolated-word, discrete-observation hidden Markov model recognition system was used for evaluation. This system was LPC based with no embellishments. In all experiments, a five state, left-to-right model was used. The system dictionary consisted of twenty highly confusable words from a speech data base formulated for recognition evaluation in diverse environments [7]. These words are also used

²Previous enhancement investigations employing colored aircraft background noise, indicated that of the spectral estimation techniques considered (maximum entropy method, maximum likelihood method, Burg's method, Bartlett's method, Pisarenko harmonic decomposition, and the Periodogram method [10, 14]), Bartlett's method produced estimates resulting in highest quality improvement for this particular distortion [3, 6].

by Texas Instruments and Lincoln Labs to evaluate recognition systems. Subsets include /go-oh-no-hello/, /six-fix/, /wide-white/, and /degree-freeze-three/. Twelve examples of each word were used, six for training, six for recognition (i.e., all tests fully open). A vector quantizer was used to generate a 64 state codebook using two minutes of noise-free training data. The twenty models employed by the HMM recognizer were trained using the forward-backward algorithm. Figure 9 presents results from five scenarios using a noise-free codebook and noise-free trained system. The 88% recognition rate clearly indicates the difficulty (confusability) of the chosen vocabulary ³. Spectral subtraction preprocessing employed three frames of magnitude averaging. The unconstrained Lim-Oppenheim approach was terminated at the third iteration. The constrained (FF-LSP:T,Auto:I) approach was terminated at the seventh iteration. Results show that recognition was reduced to chance for noisy, spectral subtraction, and Lim-Oppenheim preprocessed speech in the SNR range of (-5,0,5 dB). The constrained approach resulted in improved recognition across all SNR's considered, which is quite encouraging in light of the severe levels of noise, and difficulty of dictionary employed. An increased number of training tokens as well as a less confusable vocabulary would at the very least be required if recognition in such hostile environments is to be feasible with enhancement preprocessing. In this first set of tests, all recognition training was performed on undegraded speech. This serves to model the case of training a recognizer in advance in quiet surroundings (off-line) and using it in a noisy environment. As a final comparison, recognizer training was carried out using enhanced speech, which models training in the field. Three tests were performed using noisy and enhanced speech at a SNR of +10dB. For the noisy case, speech was coded using a noisy codebook, and recognition performed using a noisy trained HMM recognizer. Similar tests were performed for two enhancement techniques, (i.e., enhanced words coded using enhanced codebook, and tested using enhanced speech trained HMM recognizer). The results indicate that the new constrained enhancement algorithms improve recognition performance over the unconstrained Lim-Oppenheim approach. Although the scenario of training in noise, and recognizing in noise shows improvement, the recognition system is now dedicated to a specific SNR.

³On *isolated digit tasks* in quiet, the recognizer consistently scored 100% [7].

If noise characteristics or SNR should change over time, recognition performance would seriously degrade. The constraint approaches have been shown to be robust over varying SNR, and therefore should result in higher recognition rates with changing levels of SNR.

It is worth noting that although performance is poor for apparently high SNR's, the SNR computation was performed over entire words. For low energy consonantal portions, the SNR's may well be 20 dB lower; and for highly confusable word pairs (e.g., /six-fix/, /go-oh-no/), errors are understandable. A detailed analysis of the error patterns bears out this hypothesis since almost all confusions were between such pairs. For example, in one noisy speech recognition test, 43 of 61 recognition errors (70%) were caused by misclassification of distinguishing consonants, many of which were leading consonants (especially fricatives). Constrained enhancement significantly reduces these errors (e.g., one test using (FF-LSP:T,Auto:I) resulted in 16 of 21 recognition errors (with 120 test tokens) caused by misclassification of distinguishing consonants). The noise-free case itself, gave 12% errors due to the difficulty of the test set, and the small number of tokens (6) per word used for training. These results show that the new constrained techniques are valuable for recognition, especially at SNR's in the +10 to +30dB range.

4 Conclusions

The problem of enhancing speech degraded by additive white and slowly varying colored background noise was addressed. In addition, algorithm performance as a preprocessor for speech recognition was also considered. The set of enhancement algorithms presented impose inter- and intra-frame constraints on the input speech signal and were shown to be useful in enhancing speech for human listeners, and somewhat useful as preprocessing for recognition in noisy environments. Inter-frame constraints ensure more speech-like formant trajectories than those found in the unconstrained approach and thus reduce pole jitter on a frame-to-frame basis. Intra-frame constraints ensure relaxation of the iterative scheme so that overall maximum speech quality is obtained across all classes of speech. In order to increase numerical accuracy, reduce computational requirements, and eliminate inconsistencies in pole ordering across frames, the line spectral pair (LSP)

transformation of the LPC coefficients was used to implement many of the constraint requirements. The new set of constrained algorithms were shown to be effective in several domains. First, improvement in objective speech quality measures was shown. Improved LPC parameter estimation was also observed. Second, the algorithms were extended and shown to be effective on non-stationary colored noise. Third, the algorithms were shown to improve all segments of speech for both white and non-white noise. Fourth, the current algorithms have been shown to possess a consistent terminating criterion. Specifically, the optimum terminating iteration was shown to be consistent over all speech sound classes, and virtually all tested SNR's. Finally, the constrained algorithms have shown improvement as a preprocessor for speech recognition. Their ability to bring performance up to an acceptable level in SNR's between -5 and $+5$ dB is questionable. This may be due in part to the difficulty of the highly confusable test set, the small number of tokens per word used for training, and the observation that SNR's in low energy consonantal portions which discriminate confusable pairs may well be 20 dB lower. Recognition improvement in SNR's between $+10$ and $+30$ dB may be large enough to warrant enhancement preprocessing for recognition.

APPENDIX

All sentences were sampled at 8000 samples/sec.

SPEECH DATA

S1:	<i>The pipe began to rust while new.</i>	Female Speaker
S2:	<i>Thieves who rob friends deserve jail.</i>	Male Speaker
S3:	<i>Add the sum to the product of these three.</i>	Female Speaker
S4:	<i>Open the crate but don't break the glass.</i>	Male Speaker
S5:	<i>Oak is strong and also gives shade.</i>	Male Speaker
S6:	<i>Cats and dogs each hate the other.</i>	Male Speaker

References

- [1] S.F. Boll, "Suppression of Acoustic Noise in Speech Using Spectral Subtraction," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-27, pp. 113-120, April 1979.
- [2] J.R. Crosmer, "Very Low Bit Rate Speech Coding Using the Line Spectrum Pair Transformation of the LPC Coefficients," Ph.D. dissertation, School of Electrical Engineering, Georgia Institute of Technology, Atlanta, June 1985.
- [3] J.H.L. Hansen, M.A. Clements, "Enhancement of Speech Degraded by Non-White Additive Noise," Final Technical Report DSPL-85-6, Georgia Institute of Technology, Atlanta, August 1985.
- [4] J.H.L. Hansen, M.A. Clements, "Objective Quality Measures Applied to Enhanced Speech," *Proc. of the Acoustical Society of America*, 110th Meeting, C11, Nashville, Tenn., Nov. 1985.
- [5] J.H.L. Hansen, M.A. Clements, "Iterative Speech Enhancement with Spectral Constraints," *Proc. 1987 IEEE ICASSP*, pp. 189-192, Dallas, TX, April 1987.
- [6] J.H.L. Hansen, M.A. Clements, "Constrained Iterative Speech Enhancement with Application to Automatic Speech Recognition," *Proc. 1988 IEEE ICASSP*, pp. 44.S12.9.1-4, New York, NY, April 1988.
- [7] J.H.L. Hansen, "Analysis and Compensation of Stressed and Noisy Speech With Application to Robust Automatic Recognition," Ph.D. Thesis, Georgia Institute of Technology, 396 pages, July 1988.

- [8] J.H.L. Hansen, M.A. Clements, "Stress and Noise Compensation Algorithms for Robust Automatic Speech Recognition," *Proc. 1989 IEEE ICASSP*, pp. 266-269, Glasgow, Scotland, U.K., May 1989.
- [9] F. Itakura, "Line Spectrum Representation of Linear Predictive Coefficients of Speech Signals," *Journal of the Acoustical Society of America*, vol. 57, S35(A), 1975.
- [10] S. Kay, *Modern Spectral Estimation: Theory and Application*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.
- [11] J.S. Lim, A.V. Oppenheim, "All-Pole Modeling of Degraded Speech," *Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, pp. 197-210, June 1978.
- [12] J.S. Lim, Editor, *Speech Enhancement*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.
- [13] F.J. Malkin, K.A. Christ, "Human Factors Engineering Assessment of Voice Technology for the Light Helicopter Family," U.S. Army Human Engineering Laboratory Technical Report, pp. 1-20, June 1985.
- [14] S.L. Marple, *Digital Spectral Analysis with Applications*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1987.
- [15] B.R. Musicus, "An iterative technique for maximum likelihood parameter estimation on noisy data," S.M. Thesis, Massachusetts Institute of Technology, Cambridge, Mass., 1979.
- [16] J.M. Ortega, W.C. Rheinbolt, *Iterative Solutions of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.
- [17] S.R. Quackenbush, T.P. Barnwell, M.A. Clements, *Objective Measures of Speech Quality*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1988.
- [18] C.A. Simpson, "Speech Variability Effects on Recognition Accuracy Associated With Concurrent Task Performance by Pilots," Psycho-Linguistic Research Associates, Technical Report, pp. 1-15, April 1985.
- [19] F.K. Soong, B.H. Juang, "Line spectrum pair (LSP) and speech compression," *Proc. 1984 IEEE ICASSP*, pp. 705-708, San Diego, CA, March, 1984.

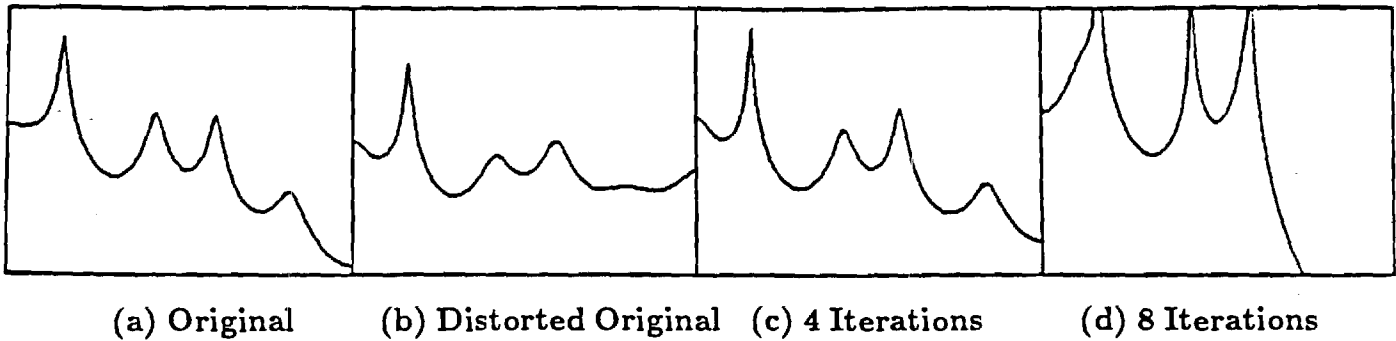


Figure 1: Variation in vocal tract response across iterations.

Sound Type	Itakura-Saito Likelihood Measure			
	Original	Lim-Oppenheim	Hansen-Clements	True LPC
Silence	1.634	1.649	0.842	0.319
Vowel	4.020	3.299	1.651	0.582
Nasal	19.814	17.656	3.968	0.324
Stop	7.261	3.979	1.099	0.435
Fricative	3.739	3.509	1.766	0.649
Glide	1.525	1.442	1.131	0.705
Liquid	9.597	4.545	0.998	0.303
Affricate	3.924	2.702	2.229	0.323
Voiced + Unvoiced	5.838	4.293	1.761	0.519
Total	4.022	3.151	1.364	0.433

Table 1: Comparison of unconstrained (Lim-Oppenheim) and inter- and intra-frame constrained (Hansen-Clements) algorithms over sound types for white Gaussian noise. SNR = +5 dB

	OBJECTIVE QUALITY MEASURE		
	Itakura-Saito	log-area-ratio	Klatt
$ \hat{\rho} $.59	.62	.74
Noisy Original	4.02	15.27	2.39
(Lim-Oppenheim)	3.15	8.78	2.19
(Hansen-Clements)	1.38	5.56	1.62

Table 2: A comparison of objective speech quality measures for noisy and enhanced speech employing the unconstrained (Lim-Oppenheim) and constrained FF-LSP: $T, Auto: I$ (Hansen-Clements) algorithms for white Gaussian noise. SNR = +5 dB, $|\hat{\rho}|$ is the average correlation coefficient between objective and subjective speech quality[17].

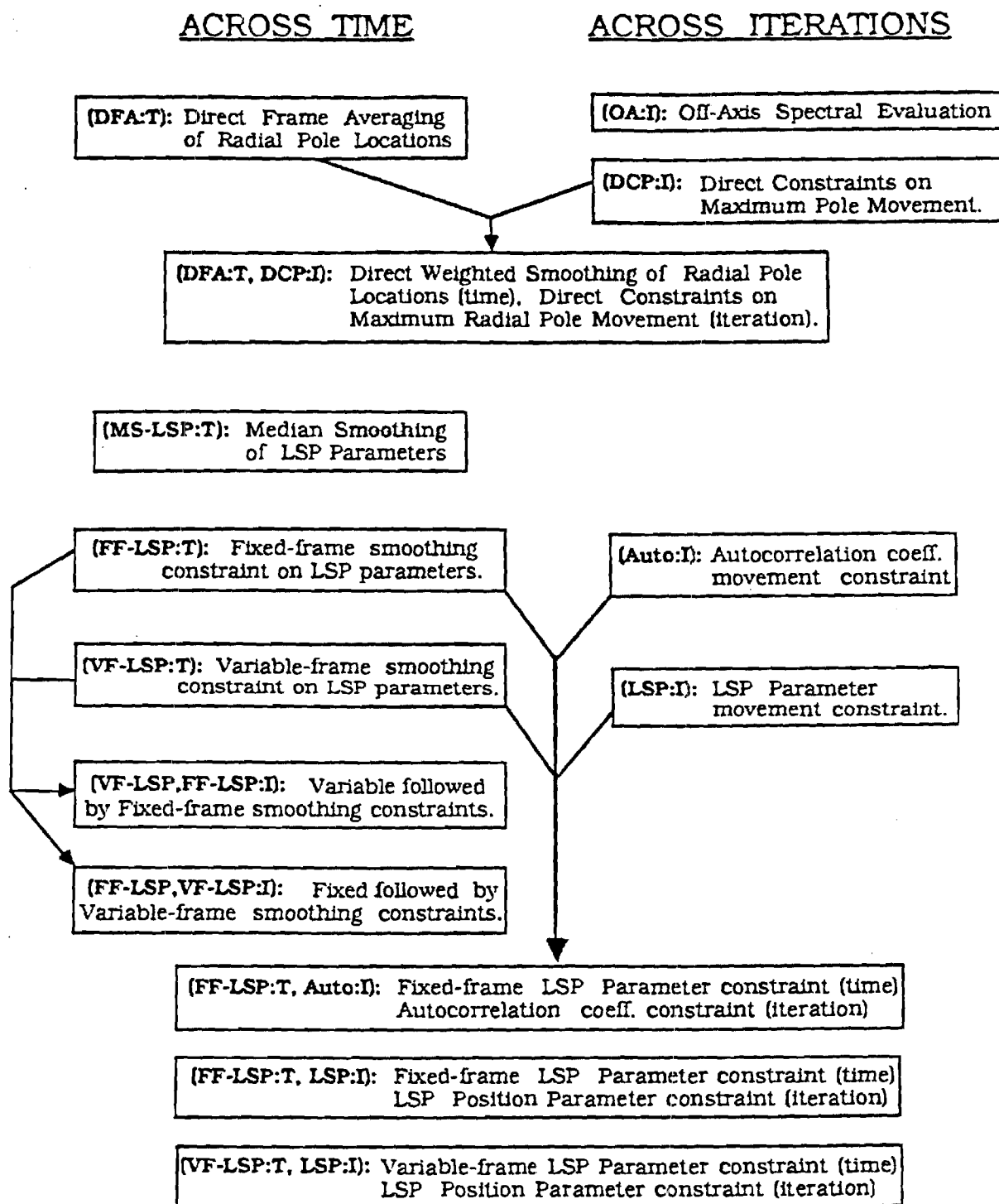


Figure 2: An overview of spectral constraints considered for the class of constrained speech enhancement algorithms.

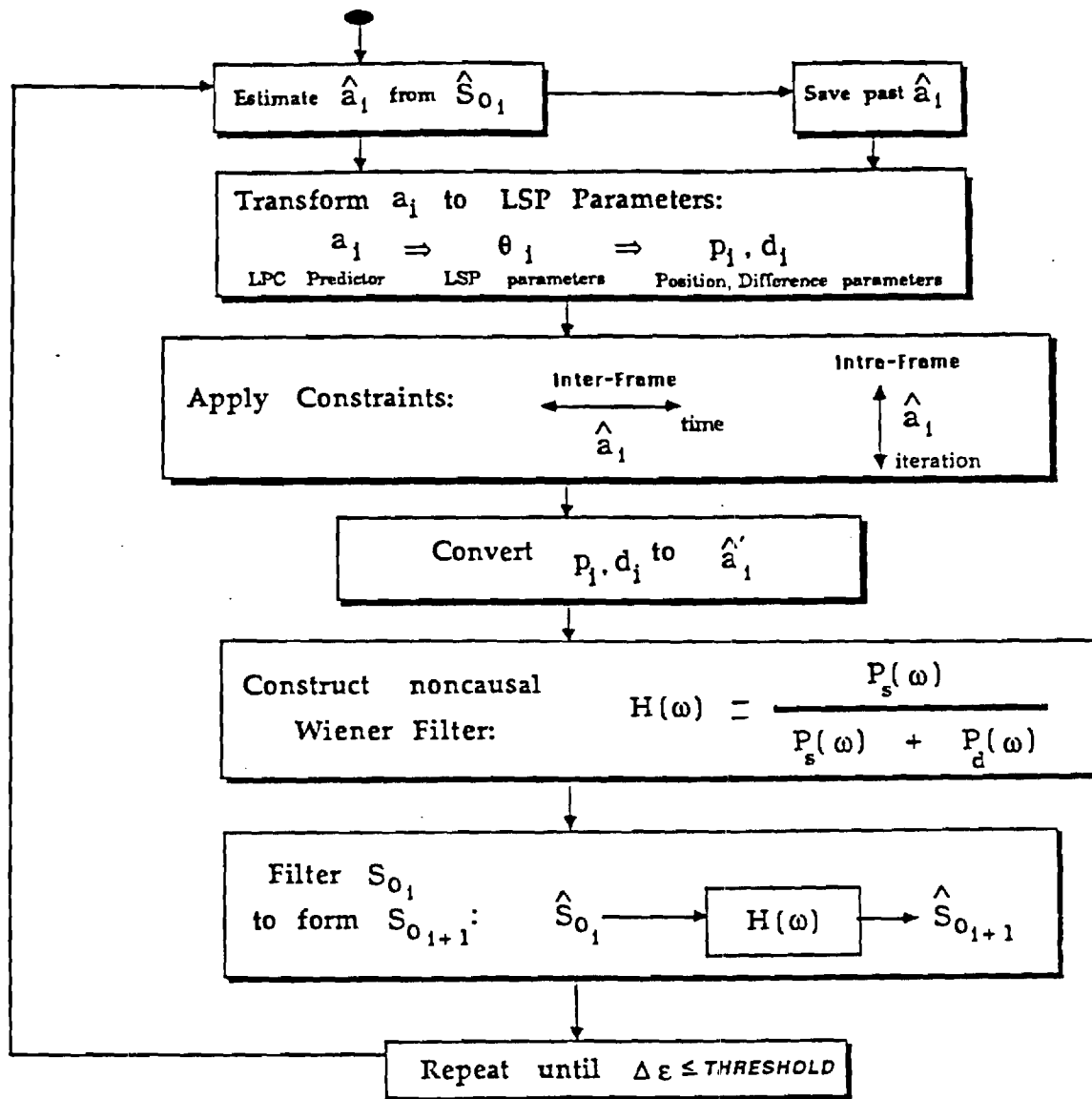


Figure 3: Framework for the new set of constrained enhancement algorithms.

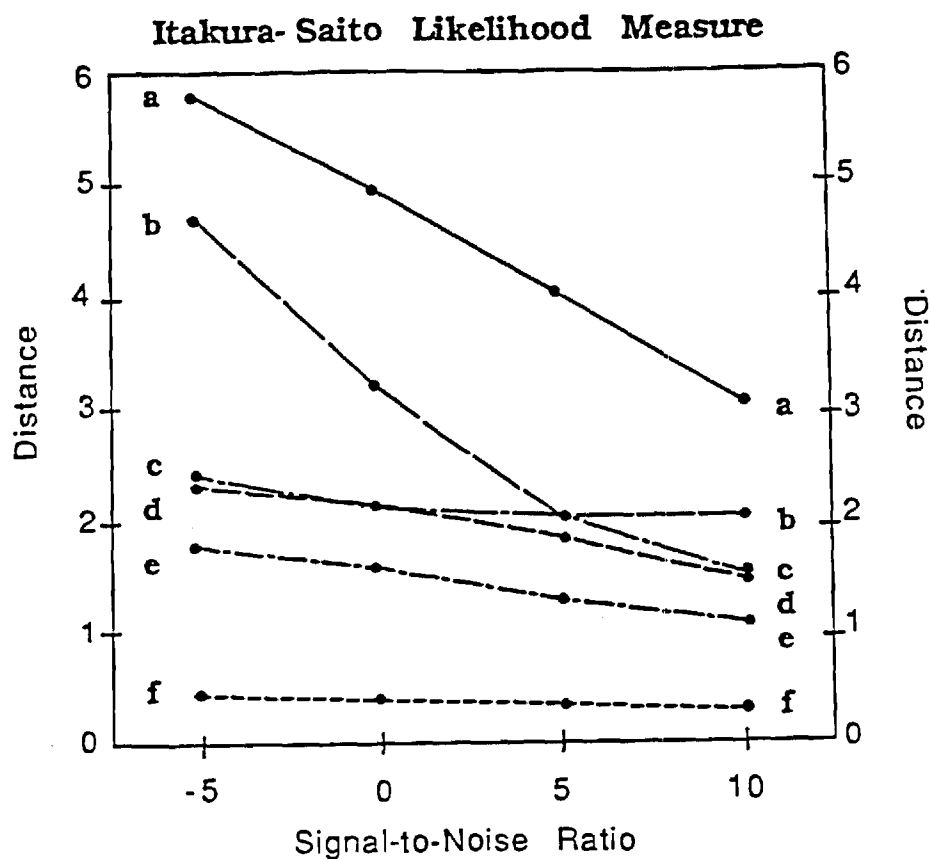


Figure 4: Comparison of constraint algorithms over SNR.

- a.) Original Distorted Speech
- b.) Inter-Frame Constraint: Variable Frame (VF-LSP:T)
- c.) Intra-Frame Constraint: Fixed Frame (FF-LSP:T)
- d.) Inter & Intra-Frame Constraints: Fixed Frame, Position (FF-LSP:T,LSP:I)
- e.) Inter & Intra-Frame Constraints: Fixed Frame, Autocorrelation (FF-LSP:T,Auto:I)
- f.) Theoretical limit: using undistorted LPC coefficients \bar{a} .

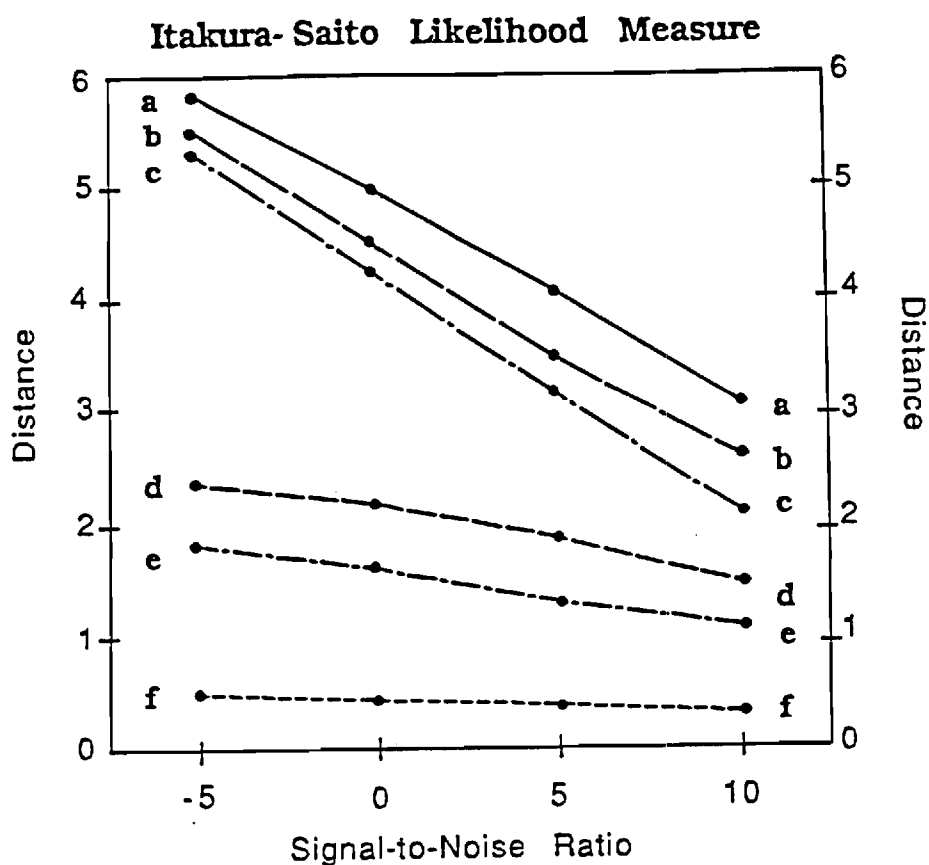


Figure 5: Comparison of enhancement algorithms over SNR.

- a.) Original Distorted Speech
- b.) Boll: Spectral Subtraction, using magnitude averaging
- c.) Lim-Oppenheim: Unconstrained Wiener filtering
- d.) Hansen-Clements: employing Inter-Frame constraints (FF-LSP:T)
- e.) Hansen-Clements: employing Inter & Intra-Frame constraints (FF-LSP:T,Auto:I)
- f.) Theoretical limit: using undistorted LPC coefficients \tilde{a} .

Sound Type	Itakura-Saito Likelihood Measure (across iterations)							
	Original	#1	#2	#3	#4	#5	#6	#7
Silence	1.634	1.615	♣1.608	1.649	1.933	3.756	20.360	49.884
Vowel	4.020	3.721	3.445	♣3.299	3.720	8.319	121.82	—
Nasal	19.814	19.154	18.416	17.656	17.009	16.593	♣15.192	15.697
Stop	7.261	6.114	4.926	3.979	♣3.822	6.889	25.515	29.694
Fricative	3.739	3.637	3.532	♣3.509	3.902	7.658	47.829	94.106
Glide	1.525	1.414	♣1.333	1.442	2.231	4.300	8.391	15.561
Liquid	9.597	8.241	6.546	4.545	2.606	♣1.676	6.381	30.001
Affricate	3.924	3.609	3.213	2.702	2.091	♣1.552	2.911	2.975
Voiced + Unvoiced	5.838	5.321	4.767	4.293	♣4.289	7.346	61.865	—
Total	4.022	3.720	3.402	♣3.151	3.271	5.795	43.457	—

Table 3: Lim-Oppenheim unconstrained speech enhancement for white Gaussian noise. Optimum perceived quality for a particular speech class in terms of objective measures is indicated by a ♣. SNR=+5dB

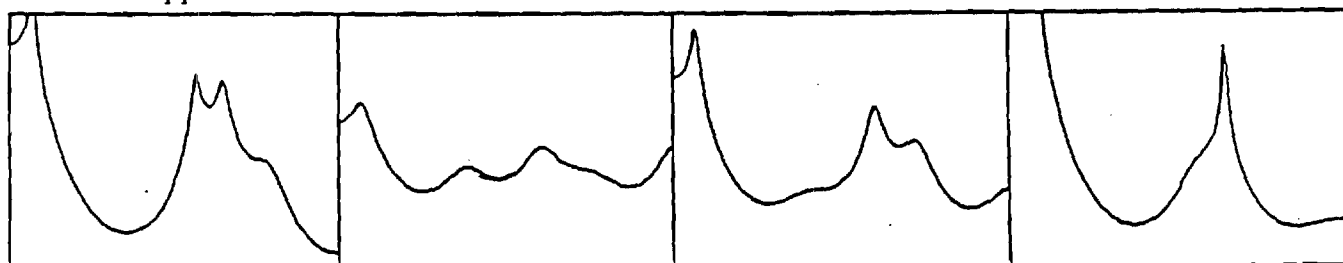
Sound Type	Itakura-Saito Likelihood Measure (across iterations)								
	Original	#1	#2	#3	#4	#5	#6	#7	#8
Silence	1.634	1.551	1.351	1.155	1.036	0.979	0.929	♣0.884	0.901
Vowel	4.020	3.319	2.865	2.394	1.863	1.677	1.571	♣1.565	1.828
Nasal	19.814	16.490	12.397	10.523	8.682	6.840	4.929	♣3.789	5.548
Stop	7.261	6.246	4.840	3.492	2.668	1.812	1.383	♣1.129	1.435
Fricative	3.739	3.432	3.027	2.612	2.245	1.948	1.729	♣1.615	1.844
Glide	1.525	1.389	1.275	1.232	1.219	1.189	1.161	♣1.153	1.217
Liquid	9.597	6.481	3.382	2.243	1.612	1.209	0.943	♣0.926	1.211
Affricate	3.924	3.722	3.447	3.117	2.806	2.598	2.472	♣2.368	3.966
Voiced + Unvoiced	5.838	4.642	3.658	3.006	2.501	2.131	1.865	♣1.740	1.953
Total	4.022	3.026	2.441	2.069	1.801	1.611	1.457	♣1.381	1.498

Table 4: Hansen-Clements Inter & Intra-frame constrained speech enhancement for white Gaussian noise. Convergence for a particular speech class in terms of objective quality is indicated by a ♣. SNR=+5dB

Constrained Enhancement Algorithm	Additive White Gaussian Noise SNR								OVERALL	
	-5 dB		-0 dB		+5 dB		+10 dB			
	Optimal Iteration using Itakura-Saito Likelihood Measure									
	Iter.	Freq.	Iter.	Freq.	Iter.	Freq.	Iter.	Freq.	Iter.	Freq.
FF-LSP:T	3	100%	3	87%	3	87%	3	100%	3	93%
			4	13%	4	13%			4	7%
VF-LSP:T	3	90%	3	85%	3	94%	3	100%	3	94%
	4	10%	4	15%	4	6%			4	6%
FF-LSP:T,Auto:I	7	100%	7	100%	7	100%	7	88%	7	97%
							6	12%	6	3%
FF-LSP:T,LSP:I	4	100%	4	100%	4	100%	4	100%	4	100%
VF-LSP:T,LSP:I	4	100%	4	100%	4	100%	4	100%	4	100%

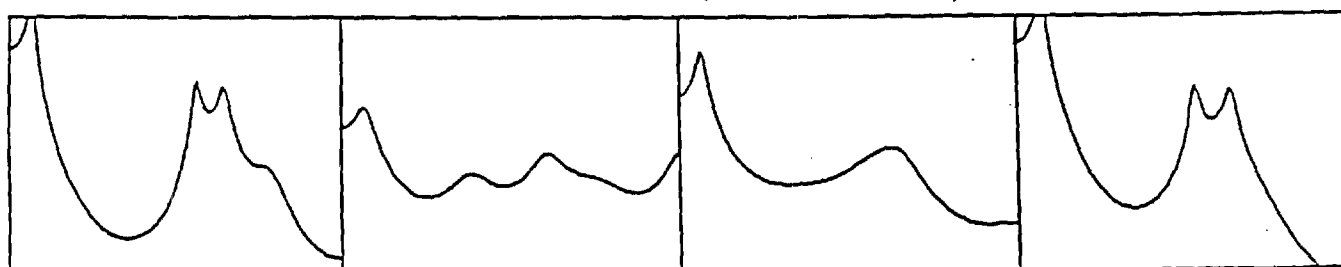
Table 5: Summary of optimal terminating iteration across SNR for AWGN.

Lim - Oppenheim: Unconstrained Enhancement



(1a) Original (1b) Distorted Original (1c) 4 Iterations (1d) 8 Iterations

Hansen - Clements: Constrained Enhancement (FF-LSP:T,Auto:I)



(2a) Original (2b) Distorted Original (2c) 4 Iterations (2d) 8 Iterations

Figure 6: Variation in vocal tract response across iterations for 1a-d) unconstrained, and 2a-d) constrained enhancement algorithms.

	<i>Itakura-Saito Measure</i>	<i>Relative Complexity (1-10)</i>	<i>Relative Computation (1-10)</i>	<i>Terminating Iteration</i>
Noisy Original	4.02			
Spectral Subtraction	3.36	2	1.5	
Lim-Oppenheim	3.15	5	3	3
(MS-LPS:T)	2.68	6	4	4
(FF-LPS:T)	1.96	7	6	3
(F-LPS:T,Auto:I)	1.36	9	10	7

Table 6: Comparison of enhancement algorithms in terms of quality, relative complexity, and relative computational resources. SNR = +5 dB, Additive white Gaussian noise distortion.

<i>Sound Type</i>	<i>Itakura-Saito Likelihood Measure</i>			
	<i>Original</i>	<i>Lim-Oppenheim</i>	<i>Hansen-Clements</i>	<i>True LPC</i>
Silence	6.63	6.33	4.32	2.03
Vowel	3.23	2.54	1.44	0.53
Nasal	4.03	3.26	2.13	0.45
Stop	1.58	1.29	0.66	0.61
Fricative	1.37	1.09	0.85	0.65
Glide	1.14	1.04	0.52	0.51
Liquid	1.22	0.55	0.22	0.18
Affricate	0.90	0.51	0.33	0.16
Voiced + Unvoiced	2.27	1.76	1.08	0.52
Total	4.15	3.86	2.74	1.17

Table 7: Comparison of generalized unconstrained (Lim-Oppenheim) and inter- and intra-frame constrained (Hansen-Clements) algorithms over sound types for slowly varying colored noise. SNR = +5 dB

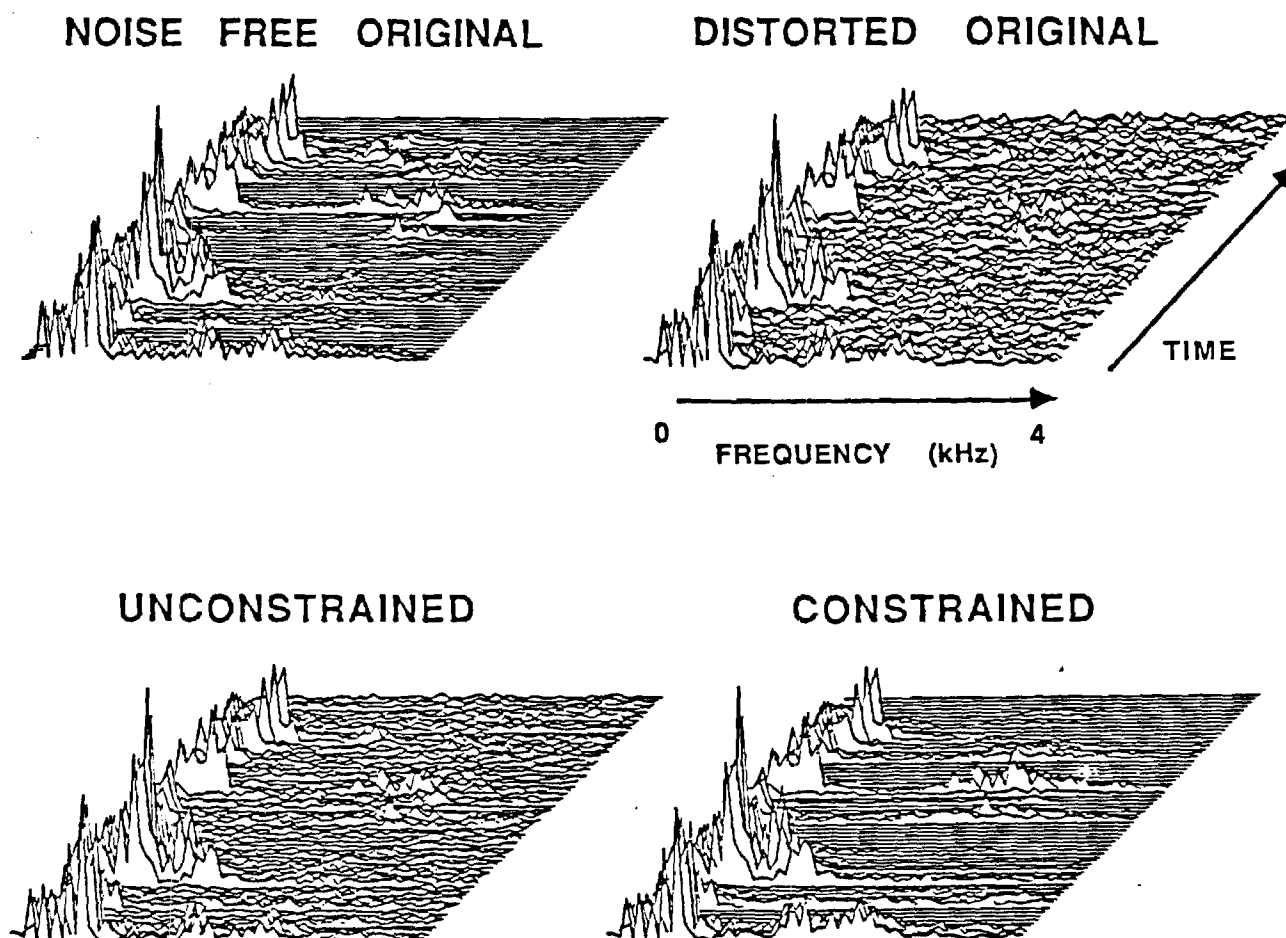


Figure 7: Time versus frequency plots of the sentence *Cats and dogs each hate the other*. The original and distorted original (additive white Gaussian noise, SNR = +5dB) are shown above. The lower left-hand plot is the response after three iterations of the unconstrained noncausal Wiener filtering approach. The lower right-hand plot is the frequency response after six iterations of an inter- plus intra-frame constrained (FF-LSP:T,Auto:I) approach.

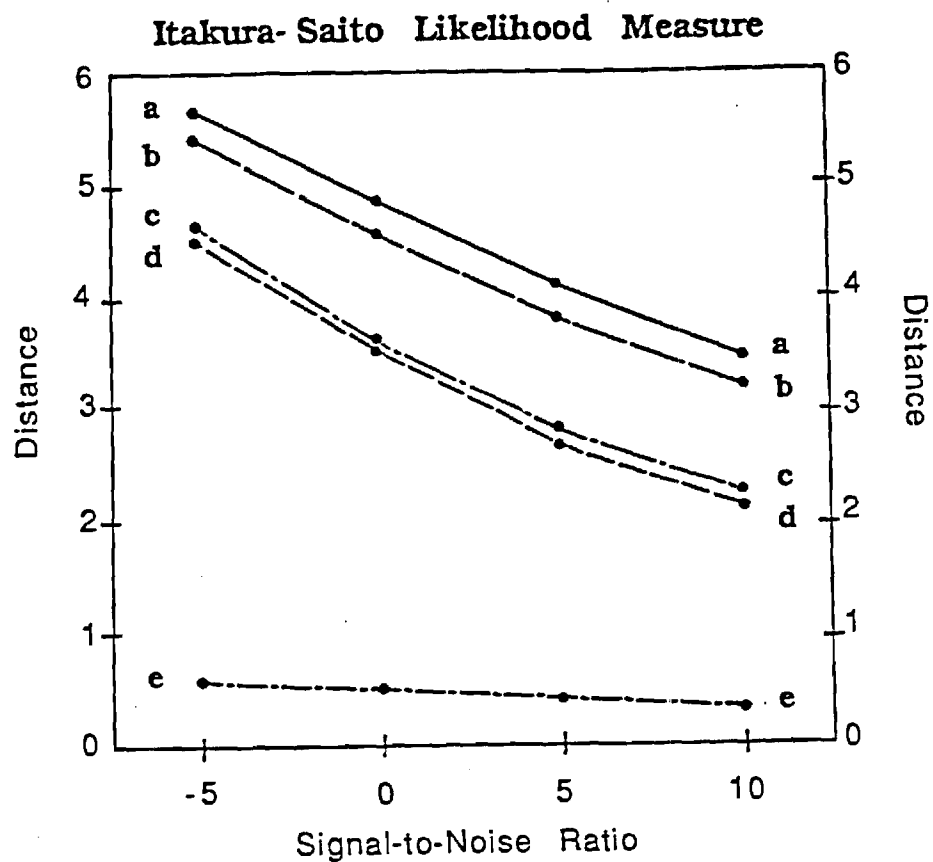
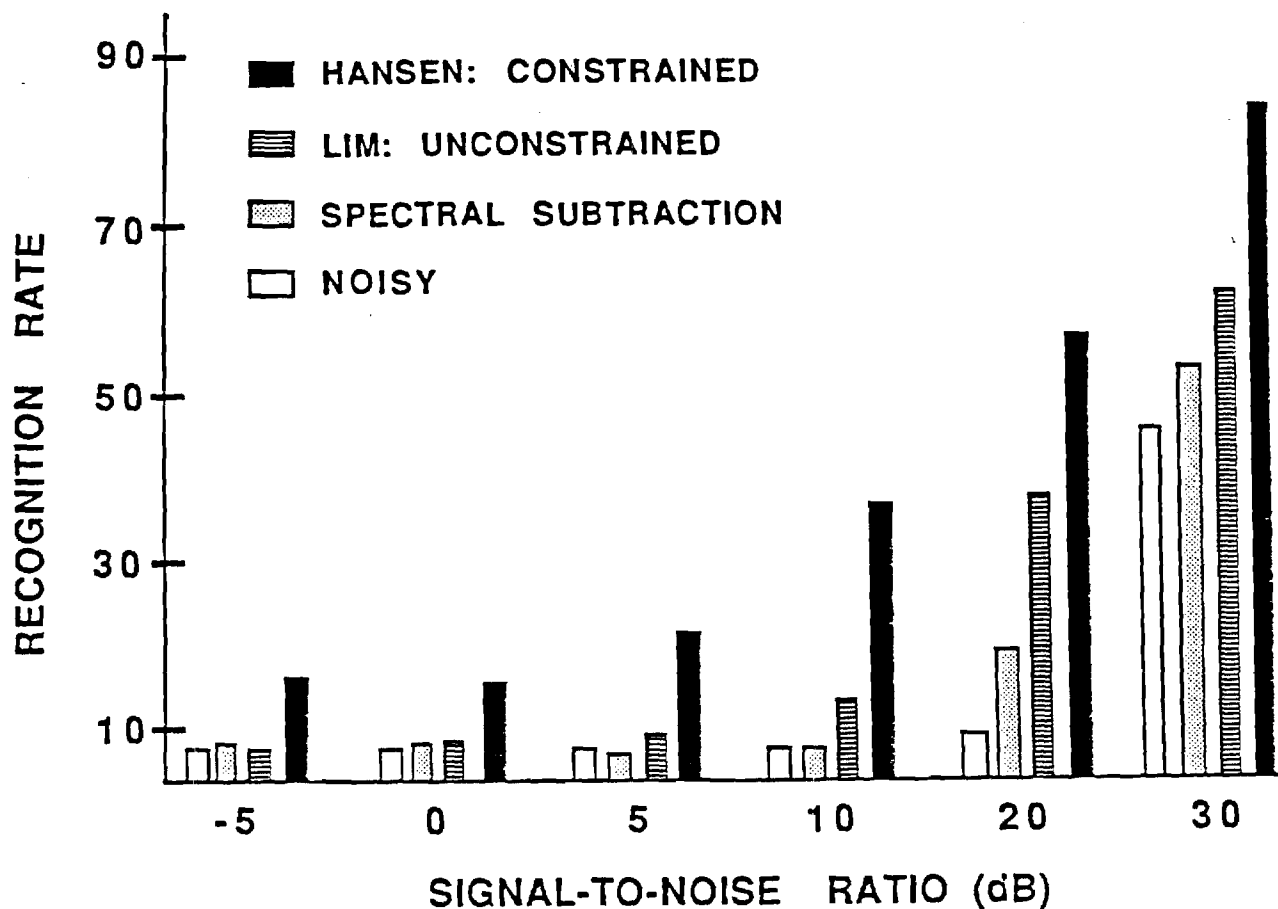


Figure 8: Comparison of inter & intra-frame constrained enhancement algorithms for colored aircraft noise over SNR.

- a.) Original Distorted Speech
- b.) Generalized unconstrained Wiener filtering
- c.) Hansen-Clements: employing Inter-Frame constraints (FF-LSP:T)
- d.) Hansen-Clements: employing Inter & Intra-Frame constraints (FF-LSP:T,Auto:I)
- e.) Theoretical limit: using undistorted LPC coefficients \bar{a} .



Condition	RECOGNITION RESULTS						
	Signal-to-Noise Ratio						
(noise-free training)	Original	-5dB	0dB	+5dB	+10dB	+20dB	+30dB
Noise-free	88%						
Noisy		5%	5%	6.7%	5%	8%	49%
Spectral Subtraction		5.8%	7.1%	5%	5.4%	20%	55%
Lim-Oppenheim		5.4%	5.8%	7.5%	12.5%	41%	64%
Hansen-Clements		15%	14%	19.5%	34.5%	59%	83%
Train & Recognize In Same Environment							
Noise-free	Noisy †	Hansen-Clements †		Lim-Oppenheim †			
88%	90%	77%		23%			

Figure 9: Recognition performance using enhancement preprocessing in additive white Gaussian noise. †SNR = +10dB